Clustering image data with a fixed embedding

1st Yan-Bin Chen
Institute of Statistical Science
Academia Sinica
Taipei, Taiwan
yanbin@stat.sinica.edu.tw

2nd Khong-Loon Tiong
Institute of Statistical Science
Academia Sinica
Taipei, Taiwan
kltiong@stat.sinica.edu.tw

3rd Chen-Hsiang Yeang
Institute of Statistical Science
Academia Sinica
Taipei, Taiwan
chyeang@stat.sinica.edu.tw

***Abstract*—Clustering unlabeled image data using deep neural network (DNN) models is under active investigation. Most existing approaches transform the data through embedding operations and cluster the embedded data, and the embedding is learned to fit the data. In some applications, the embedding model is explicitly given due to the concerns of generalizability, transferability, privacy and security. Despite rapid progress in self-supervised learning, clustering data with a fixed embedding is rarely explored. We propose an Merge & Expand (ME) algorithm to cluster image data using a fixed embedding and a DNN classification model. ME achieves a comparable level of accuracy with some state-of-the-art algorithms. It further demarcates the "clean" and "unclean" images where their geometric relations in the embedded space are compatible and incompatible with their cluster structure respectively. Finally, we validate ME with three datasets and discuss its potential extension.**

***Index Terms*—Clustering, CNN, Embedding**

I. Introduction

Extracting relevant information from unlabeled data is of great importance as the vast majority of existing data lack labels. Clustering is a representative topic in this domain. Metric-based clustering algorithms are often sensitive to the perturbations of the input data which do not alter their semantic content [1]. By contrast, deep neural networks (DNNs) extract features invariant with these perturbations, thus generate more robust outcomes. Yet they require proper loss functions which do not use the label information. Recently, an increasing number of self-supervised learning algorithms obtain supervisory information from the data and adopt DNNs to tackle unsupervised learning problems including clustering [2]. Self-supervised clustering algorithms of image data roughly fall into two categories. Sequential (or representation learning) methods first generate embeddings or feature representations of the unlabeled data and then employ clustering or classification algorithms to the embedded data [3]–[5]. The embedded feature vectors are generated by DNNs to optimize certain tasks without requiring class labels, such as predicting the patch context [6], solving jigsaw puzzles [7], or colorizing images [8]. Joint (or end-to-end learning) methods unite embedding learning and clustering in training the same DNNs. These neural networks differ in their architectures and loss functions, which often require images undergoing perturbations to cluster together [9]. Combinations of both sequential and joint methods are also proposed in several studies [10], [11].

Despite the rapid progress in self-supervised image clustering, several major issues are overlooked in previous studies. The embedding generated by customized algorithms may better fit the data but is inadequate or unavailable in some contexts. The embedding learned from a small data may be difficult to generalize as it often contains the specific features tied to the data but irrelevant to the semantic content of the underlying class labels. For instance, one might use the background colors and textures to separate the images of wild and domestic animals. In contrast, the embedding generated by pre-trained models can overcome data limitation and become more transferable across different tasks. Examples of pre-trained DNN models include VGG16 [12] for image data and BERT [13] for language data. Furthermore, in some applications, the raw data may not be directly accessible for privacy or security concern or for its sheer size. Rather than releasing the raw data, the providers may incur standard embedding (such as VGG16) or dimension reduction (such as PCA and t-SNE) operations to the raw data and distribute the processed data. Clustering image data with a fixed embedding is hence increasingly important but not well explored so far. In addition, a general embedding may be only partially aligned with the underlying cluster structure. It is important to chart the limitation of the fixed embedding, i.e., to demarcate the data points whose underlying cluster structure is compatible or incompatible with their geometric relations in the embedded space. This is a challenging problem since the class labels are not given.

In this work, we present a self-supervised learning algorithm to cluster image data and simultaneously address the aforementioned issues. Like most sequential methods, we project the data onto the embedded space and cluster the embedded feature vectors. However, unlike other methods, we employ standard embedding operations, simple convolutional neural network (CNN) architectures and straightforward cross entropy loss functions for classification, but achieve a comparable level of accuracy with some state-of-the-art algorithms. Furthermore, the algorithm demarcates the "clean" and "unclean" images where the geometric relations in the embedded space are compatible and incompatible with the underlying cluster structure respectively.

This work is supported by the funding from Ministry of Science and Technology, Taiwan, No: 108-2118-M-001-001-MY2, and seed grant from Institute of Statistical Science, Academia Sinica, Taiwan.

II. METHOD

A. Overview

The proposed algorithm, named "Merge & Expand" (ME), comprises 5 phases, as illustrated in Fig. 1. In the preparation phase, Phase 0, the raw data are transformed into feature vectors with a pre-selected embedding operation. In Phase 1, the embedded data are grouped into small and tight clusters, termed "regions", by using a metric-based clustering algorithm. We apply spectral clustering in this work. In Phase 2, we select a number of regions, termed "seed regions" that likely cover all the underlying class labels. Seed regions should be tight, so that it is safe to assign members of a region with the same pseudo label; and distinct from each other so that they cover all the underlying class labels. In Phase 3, we merge seed regions and assign members of each merged seed region to a distinct pseudo label through the 7-layer LeNet-5 CNN architecture [14]. In Phase 4, using the merged seed regions as the initial training data, we iteratively apply the CNN classifier to predict the labels of the whole data and expand the training data by including the regions containing the dominant predicted labels. Iteration continues until no regions are included in the training data. Cluster assignments are determined by the final classification outcomes. The regions which are not included in the training data comprise heterogeneous predicted labels, and their members are marked as unclean images.

B. Merge & Expand (ME) Procedures

1) Phase 0: Embedding the raw data: The embedding operation is selected from well-known dimension reduction algorithms or pre-trained DNN models. A standard dimension reduction projection is the t-Distributed Stochastic Neighbor Embedding (t-SNE) projection that preserves the conditional probabilities specifying the inter-image distances [15]. The raw image data, which is a vectorized matrix of pixel values, are projected onto low dimensional vectors. A well-known transfer learning DNN model of images is the VGG16 model pre-trained from 14 million images of the ImageNet database belonging to 1000 categories [16]. The raw image data are fed into the model and the 1000-component feature vectors before the final decision step are reported. An embedded feature vector thus quantifies resemblance of a raw image with each of the 1000 categories. Other off-the-shelf embedding algorithms can also be used.

2) Phase 1: Clustering the embedded data into regions: The embedded data are grouped into regions – small and tight clusters – using a metric-based clustering algorithm. Since the class labels of data are not given, we can only control the homogeneity of the underlying class labels by the size and geometry of the region members. We employ the spectral clustering algorithm [17] to subdivide the embedded data into n_R (200 in our work) regions. The following notations are introduced. Denote $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$ the input data of N images, $R(\boldsymbol{x}) \in \{1, \cdots, n_R\}$ the region label of each data point, and $r_i \equiv \{\boldsymbol{x} \in \boldsymbol{X} : R(\boldsymbol{x}) = i\}$ the members of region i. The input data could be the raw data or the embedded data, according to the applications. In this phase, we further consider periphery members, informative markers, and use spectrum clustering to generate initial regions. The algorithm of it is depicted in supplement [1].

3) Phase 2: Selecting seed regions: To apply CNN to cluster the embedded data, we need to acquire a reliable training data and assign its pseudo labels. To fulfill this goal, we select the seed regions – a minimal set of regions that cover all the underlying class labels – as the initial training data. Since the class labels are not given, we alternatively seek a minimal set of regions that yield a stable partition of the embedded data. The algorithm starts with two regions which are far apart and attract similar numbers of data points, and then iteratively adds the regions which are distinct from all existing seed regions. Define $d(r_i, r_j) \equiv \frac{1}{|r_i||r_j|} \sum_{\{\boldsymbol{x} \in r_i, \boldsymbol{x}' \in r_j\}} d(\boldsymbol{x}, \boldsymbol{x}')$ the average distance between members of regions r_i and r_j in the embedded space. Given a collection of initial seed regions $S = \{r_1, r_2\}$, the next seed region is selected by a max-min criterion:

$$\hat{r} = \arg \max_{r \notin S \cup F} \min_{r' \in S} d(r, r'), \tag{1}$$

where the candidate regions exclude the existing seed regions S and the filtered regions F. A filtered region in F may comprise members of mixed class labels or the ones which do not share class labels with their neighboring regions. The max-min criterion iteratively selects the region which is the most distinct from all existing seed regions.

In addition, the newly selected seed region is skipped if it can replace one of the existing seed regions and incur similar partitions of the embedded data. Seed region selection stops when 5 consecutive newly selected seed regions are skipped. The algorithm of selecting seed regions is reported in supplement.

4) Phase 3: Merging seed regions: Despite their distinctiveness, multiple seed regions may share the same underlying class labels because regions are tight clusters in the embedded space. For instance, images of the same digit written with different tilted angles or images of the same class of objects with different colors are likely represented by distinct seed regions. To give more precise pseudo label assignments to the training data, it is necessary to merge the seed regions that likely share the same underlying labels. We reason that relative similarity between seed regions can be captured by DNN classification outcomes using combinations of the seed regions as training data because well-designed DNNs may give robust predictions against various perturbations. For example, the training data combining red and blue cars in the same class has better generalization power than the training data combining red cars and birds in the same class since the former share more common features than the latter.

a) Co-contribution score $\boldsymbol{M}$[1]: As the seed region set S is prepared from Seed Region Selection Algorithm in the main paper, we build n_r CNN classifiers with K seed regions as

[1] http://staff.stat.sinica.edu.tw/chyeang/ME/MESupplement.pdf

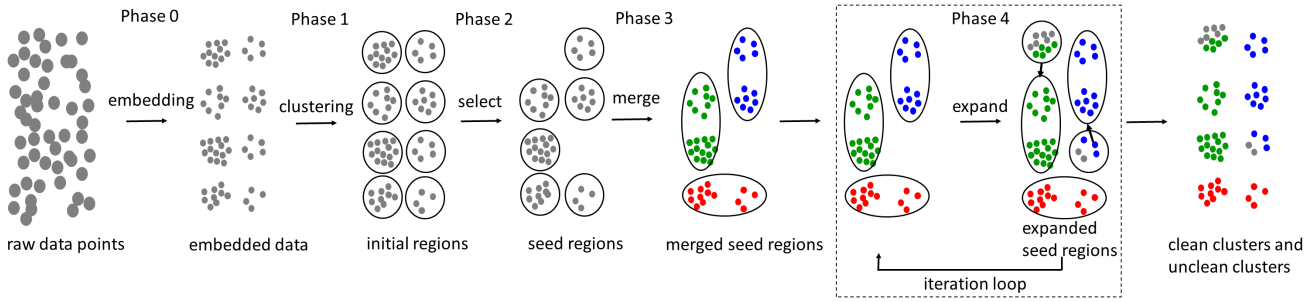


Fig. 1: An illustration of the ME algorithm. Gray dots denote the images which have not yet been assigned to pseudo labels (phases 0-2) or are not confidently classified (phase 4). Colored dots denote the images which are assigned to pseudo labels.

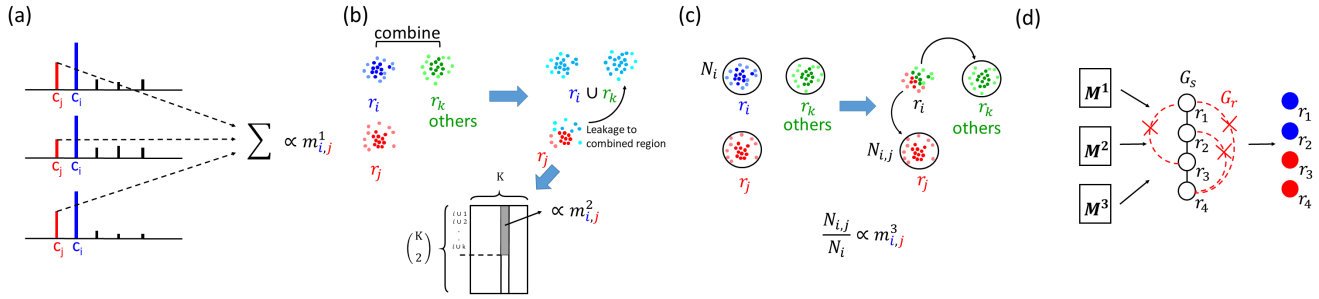


Fig. 2: Illustration of three pairwise scores in Phase 3. (a) Co-contribution. Each "spectrum" denotes the CNN outputs of a data point which is labeled to class i (thus possessing the highest output value in c_i as shown in blue spikes). m^1_{ij} is proportional to the sum of output values in c_j component (as shown in red spikes) over the data points labeled to c_i. (b) Leakage. Combine each pair of seed regions into one class $i \cup k$. A number of data points originally assigned to class j (red dots before combination) will "leak" to the combined class $i \cup k$ (light blue dots in region r_j after combination). Construct a $\binom{K}{2} \times K$ matrix to count the leakages from each class j to all combined classes (the marked column in the matrix). m^2_{ij} quantifies the leakages from j to the combined classes containing i (the shaded entries in the column) with respect to the leakages from j to all the combined classes (the entire column). (c) Replacement. Remove each seed region r_i from the training data and rebuild a $(K-1)$-class CNN. m^3_{ij} quantifies the fraction of data points which are originally assigned to i (blue dots in the left) and are re-assigned to j after removal (uncircled red dots in the right). (d) Integration. Similarity graph G_s (solid black lines) and dissimilarity graph G_r (dashed red lines) of seed regions are constructed according to $\boldsymbol{M^1}$, $\boldsymbol{M^2}$ and $\boldsymbol{M^3}$. Partition the seed regions into a small number of merged seed regions that respect both G_s and G_r.

the training data of K classes ($n_r = 5$ in this work). The n_r sets of result from CNN may not give identical prediction outcomes as the stochastic gradient descent (SGD) parameter estimation operates randomly. The co-contribution score m^1_{ij} is proportional to the sum of the output values of class j over the data points with consensus label i:

$$\boldsymbol{M}^1 = [m^1_{ij}] \equiv \frac{1}{|T_i| \cdot n_r} \sum_{\boldsymbol{x} \in T_i} \sum_{l=1}^{n_r} f^l_j(\boldsymbol{x}), \tag{2}$$

where T_i denotes the set of data points with consensus label i, $f^l(\boldsymbol{x})$ the round l CNN outputs of data point $\boldsymbol{x}$ over K classes, and $f^l_j(\boldsymbol{x})$ its j^{th} component, as shown in Fig. 2(a). The score is normalized by the size of T_i times the number of CNN rounds n_r. Intuitively, m^1_{ij} is high if the images closest to seed region i also possess high CNN outputs pertaining to seed region j.

b) Leakage score $\boldsymbol{M}^2$: We combine each pair of seed regions into one class in the training data and build $\binom{K}{2}$ predictors with $(K-1)$-classes accordingly. Denote $y(\boldsymbol{x})$ as the predictor of data point $\boldsymbol{x}$. For each pair (i,k) of seed regions, we consider the original K-class predictor $y_0(\boldsymbol{x})$ and a $(K-1)$-class predictor $y_{i\cup k}(\boldsymbol{x})$ merging seed regions i and k in the same class. By comparing $y_0(\boldsymbol{x})$ with each $y_{i\cup k}(\boldsymbol{x})$, we construct a $\binom{K}{2} \times K$ leakage matrix $L_{i\cup k,j}$ to count the number of data points which are assigned to class j according to $y_0(\boldsymbol{x})$ and to the combined class $i \cup k$ according to $y_{i\cup k}(\boldsymbol{x})$:

$$L_{i\cup k,j} \equiv \Big|\{\boldsymbol{x} : y_0(\boldsymbol{x}) = j,\ y_{i\cup k}(\boldsymbol{x}) = (i \cup k)\}\Big|. \tag{3}$$

Intuitively, if seed regions i and j share the same underlying class label, then a considerable number of data points originally assigned to pseudo class j will leak to the combined pseudo class i with an arbitrary third pseudo class k, as shown in Fig. 2(b). Consequently, on the j^{th} column of L, the high leakage counts should be enriched in the combined pseudo classes containing seed region i. We calculate the leakage score m_{ij}^2 as the p-value of this enrichment in (4):

$$\boldsymbol{M}^2 = [m_{ij}^2] \equiv P_{W\left(L(R_i,j),\, L(R_{1:K},j)\right)}, \tag{4}$$

where R_i denotes the rows in L corresponding to the pairs containing seed region i and $R_{1:K}$ all the rows in L, $W(L(R_i,j),\ L(R_{1:K},j))$ the Wilcoxon rank sum test [18] of $L(R_i,j)$ against the background $L(R_{1:K},j)$, and $P_{W(L(R_i,j),\, L(R_{1:K},j))}$ its p-value. The value m_{ij}^2 is low if the high leakage of the data points originally assigned to class j is concentrated in the combined classes containing class i.

c) Replacement score $\boldsymbol{M}^3$: We remove each seed region from the training data and build K $(K-1)$-class predictors accordingly. The replacement score m_{ij}^3 is the fraction of data points which are reassigned from pseudo class i in the original K-class predictor $y_0(\boldsymbol{x})$ to pseudo class j in the $(K-1)$-class predictor $y_{-i}(\boldsymbol{x})$ with seed region i removed from the training data:

$$\boldsymbol{M}^3 = [m_{ij}^3] \equiv \frac{\left|\{\boldsymbol{x} : y_0(\boldsymbol{x}) = i,\ y_{-i}(\boldsymbol{x}) = j\}\right|}{\left|\{\boldsymbol{x} : y_0(\boldsymbol{x}) = i\}\right|}. \tag{5}$$

Intuitively, if seed regions i and j share the same underlying class labels, then pseudo class j will replace pseudo class i when the latter is removed from the training data, thus yields a high replacement score. This replacement is displayed in Fig. 2(c).

d) Integration of $\boldsymbol{M}^1$, $\boldsymbol{M}^2$, and $\boldsymbol{M}^3$: We integrate the three pairwise score matrices $\boldsymbol{M}^1$, $\boldsymbol{M}^2$, and $\boldsymbol{M}^3$ to merge seed regions. In brief, two undirected graphs G_s and G_r of seed regions are constructed from the three score matrices. G_s specifies similarity of seed regions where two nodes are adjacent if one node is a top-ranking partner of the other in terms of each score in both directions, and the pairwise scores surpass certain threshold values. G_r specifies dissimilarity of seed regions where two nodes are adjacent if their leakage score m_{ij}^2 (rank sum p-value) is insignificant. The undirected graph of seed regions is exhibited in Fig. 2(d). We then enumerate all partitions of seed regions respecting the relations in G_s (so that nodes in distinct connected components are not merged together) and G_r (so that adjacent nodes are not merged together), and identify the valid partition that minimize loss functions pertaining to the number of singletons, the sum of the reduced rank scores derived from the pairwise scores, and the sum of the pairwise scores. The output of Phase 3 is a partition of seed regions into a smaller number of merged seed regions.

5) Phase 4: Expanding training data, assigning cluster memberships, and demarcating clean and unclean images: Since seed regions constitute a minimal set of regions that likely cover the underlying class labels, the training data containing seed regions alone may be too small to generalize to the whole data. To improve classification quality, we iteratively include the regions with homogeneous predicted class labels in the training data. Initially the training data includes only the merged seed regions. In each iteration, we build a CNN classifier with the training data and predict the class labels of the all remaining "test data". Homogeneity of predicted class labels in a region is quantified by the fraction of member images with the dominant (the most frequent) predicted class label. We select the regions outside the training data whose homogeneities exceed a threshold value, include the members with the dominant predicted class labels of selected regions in the training data, and update the CNN classifier accordingly. Iteration stops when no more regions are added to the training data. The predicted class labels of all images in the final iteration are their cluster assignments. The regions not included in the training data in the final iteration comprise members of heterogeneous predicted class labels. The cluster assignments of members in these regions are less confident since they are incompatible with the proximity relations in the embedded space; i.e., neighboring images in the embedded space possess different cluster assignments. We thus mark these regions as "unclean". In contrast, the regions included in the training data are marked as "clean" since their cluster assignments are compatible with their proximity relations in the embedded space.

III. Results

We validate the ME algorithm on two common datasets for machine learning – MNIST and CIFAR-10 – and compare the accuracies of several unsupervised/self-supervised learning algorithms on these datasets.

A. MNIST

Clustering MNIST, which is 60000 images of 10 handwritten digits, is considered an easy task as many recent unsupervised learning algorithms give highly accurate outcomes. Projections of the MNIST data with t-SNE clearly separate the 10 digits [15]. Indeed, t-SNE projections can lift the accuracy rates of k-means and spectral clustering algorithms from 57.2% and 69.6% in the raw data (28×28 pixels) to 73.3% and 96.6% in the embedded data (3D projections). The ME algorithm clustering outcome achieves 91.5% accuracy rate (Table I, column 1).

B. MNIST-TRAN

Despite its outstanding capability to separate digits, t-SNE is not robust against simple perturbations such as translations and rotations. To demonstrate that ME can still accurately cluster images undergoing such perturbations, we augment the MNIST data by adding 10000 images with translations. 500 images of each digit are randomly selected, and each

TABLE I: Overall accuracies of several clustering algorithms on three datasets. All but the first two and last ones are copied from [11], [19]. For k-means and spectral clustering (SPECTRAL CLT), accuracy rates on MNIST raw data and t-SNE embedded data are reported; accuracy rates on MNIST-TRAN t-SNE embedded data are reported; accuracy rates on CIFAR-10 raw data and VGG-16 embedded data are reported. The accuracy rates of most methods on MNIST-TRAN data are not reported. The clean and unclean accuracies by ME are reported at the last row. DEEP CLT2018 refers to DeepCluster2018.

METHOD	(1)MNIST	(2)MNIST-TRAN	CIFAR-10
K-MEANS	57.2/73.3	66.7	22.9/52.7
SPECTRAL CLT	69.6/96.6	70.8	24.7/59.8
TRIPLETS	52.5	N/A	20.5
AE	81.2	N/A	31.4
GAN2015	82.8	N/A	31.5
JULE2016	96.4	N/A	27.2
DEC2016	84.3	N/A	30.1
DAC2017	97.8	N/A	52.2
DEEP CLT2018	65.6	N/A	37.4
ADC2018	99.2	N/A	32.5
IIC	99.2	N/A	61.7
TSUC+RUC	N/A	N/A	82.9
SCAN+RUC	N/A	N/A	89.5
ME	**91.5**	**91.2**	**69.2**
			↓
ME CLEAN ; UNCLEAN			**81.5; 35.6**

image undergoes translations with two fixed displacement vectors (total: $(5500 + 500 + 500) \times 10$). We name this dataset MNIST-with-translated-data (MNIST-TRAN). The performance of both k-means and spectral clustering algorithms substantially deteriorates on MNIST-TRAN data, as accuracy rates on the embedded data (by t-SNE) drop to 66.7% and 70.8% for k-means and spectral clustering, respectively.

We apply ME to the MNIST-TRAN data. The 25 seed regions are not robust against translations as they are derived from clusters of raw images (Fig. 3(b)). For instance, digit 9 images undergoing two different translations clearly separate from the untranslated digit 9 images and each other, but are closer to the images of other digits (Fig. 3(a)). By training the CNN classifiers with the raw data rather than the 3-dimensional embedded vectors, the three pairwise scores give rise to merged seed regions which are robust against translations (Fig. 3(c)). The clustering accuracy rate 91.2% remains almost invariant with the original MNIST data (Table I, column 2). The accuracy rates of other methods are not reported on the MNIST-TRAN data.

C. CIFAR-10

CIFAR-10 is an image dataset consisting of 60000 32×32 color images in 10 classes, with 6000 images per class. We embed CIFAR-10 data by feeding it to the VGG16 model. The embedded data of a CIFAR-10 image is the VGG16 output prior to the decision step, which is a 1000-component vector. This transformation greatly amplifies information about the underlying class labels as all CIFAR-10 classes are either encompassed or closely related in the 1000 categories in VGG16. Indeed, the accuracy rates lift from 22.9% in the raw data to 52.7% in the VGG16 embedded data for k-means, and from 24.7% in the raw data to 59.8% in the VGG16 embedded data for spectral clustering. The overall clustering accuracy rate is 69.2% (Table I, column 3). It is considerably lower than those of the two best algorithms (SCAN+RUC and TSUC+RUC; [11]) but substantially higher than all other methods.

Although ME does not achieve the highest accuracy rate compared to state-of-the-art algorithms, it possesses a unique feature of demarcating the images with reliable and unreliable cluster assignments, namely clean and unclean images. There are 31879 clean and 28121 unclean images in the CIFAR-10 data. Surprisingly, these two sets of images exhibit drastically different clustering accuracy rates: 81.5% for clean images and 35.6% for unclean images. The 69.2% overall accuracy is hence an aggregate from the sets of images with highly and poorly reliable cluster assignments.

The cleanness of a region is determined by heterogeneity of the predicted class labels among its members. We suspect that this property holds for the true class labels as well. Fig. 4(a) and 4(b) display examples of clean and unclean images respectively. The clean image – the center image of Fig. 4(a) – and its 8 nearest neighbors in the embedded data – the 8 images in its periphery – all share the same class label of horses. In contrast, the unclean image – the center image of a dog in Fig. 4(b) – and its 8 nearest neighbors all have distinct class labels. These examples are not exceptional. Fig. 4(c) shows the distributions of the numbers of consistent neighbors (neighbors sharing the same true class labels) in the 10 nearest neighbors among clean (blue), unclean (red) and all (black) images. Most clean images share the same true class labels with most of their neighbors. In contrast, the true class labels of unclean images are poorly correlated with the true class labels of their neighbors for the numbers of their consistent neighbors are uniformly distributed. The distribution of numbers of consistent neighbors among all images is the superposition of those of clean and unclean images.

IV. DISCUSSION

In this study, we propose a Merge & Expansion (ME) algorithm to combine both metric-based clustering and deep supervised learning algorithms to cluster unlabeled image data. The ME algorithm does not give the most accurate cluster assignments compared to several state-of-the-art self-supervised learning algorithms, but is superior to most other methods. More importantly, ME provides several unique benefits and features not found in other methods: (1) it clusters data with a fixed embedding (or no embedding) model rather than learning the embedding from the data, (2) it identifies clean and unclean images where the geometric relations in the embedded space are compatible/incompatible with the underlying cluster structure, (3) it employs simple, off-the-shelf CNN architectures

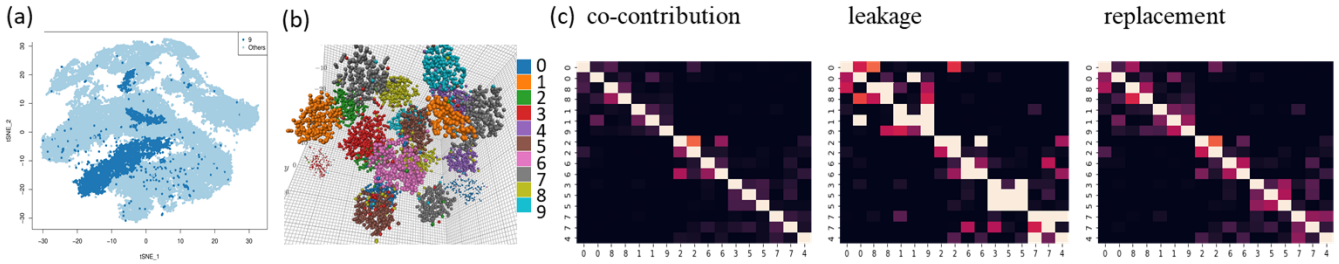
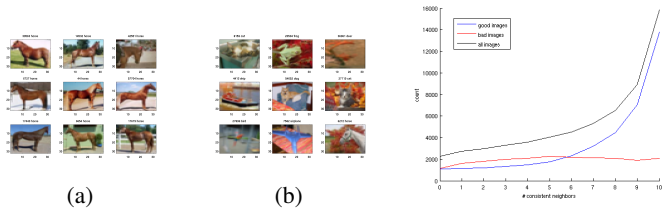


Fig. 3: (a). MNIST-TRAN t-SNE 2D projections. Digit 9 images are highlighted (dark blue dots). One large cluster in the lower subspace and two small clusters in the upper subspace denote the untranslated digit 9 images and two sets of translated digit 9 images respectively. (b). t-SNE 3D projections of seed regions on MNIST-TRAN dataset. Tiny points refer to the translated regions. (c). Heat maps of 3 pairwise scores of seed regions on MNIST-TRAN.



(a) (b) (c)

Fig. 4: (a). An example of clean image. (b). An example of unclean image. (c). Distributions of numbers of consistent neighbors of clean, unclean and all images in CIFAR-10 data.

and loss functions instead of customized DNN models, (4) the number of clusters is determined by the seed region merging procedures rather than being explicitly given.

Using a fixed embedding model is a double-bladed sword. On the positive side, a fixed embedding is less tied to specific data and is thus more generalizable to other tasks, is amenable for transfer learning, and allows users to perform clustering without accessing the raw data. On the negative side, the performance is compromised as the geometry in the embedded space is unlikely fully aligned with the cluster structure. For instance, two CIFAR-10 classes – dogs and cats – have quite disparate representation coverages in the VGG16 model. The 1000 categories include many dog varieties but only a few cats/felines. Therefore, cat images in CIFAR-10 may have high values in the dog variety components of the embedded vectors and are likely mislabeled as the dog class. Despite the limitation of a fixed embedding, our method may circumvent this limitation by pointing out the data points where the embedded data can better predict the underlying cluster structure or not.

The ME algorithm can be improved and extended. To enhance the explanatory power of a fixed embedding it is desirable to build a hybrid model for embedding using both pre-trained networks and data-specific DNNs, analogous to the transfer learning models in classifications [20]. Furthermore, the ME framework can be extended to data modalities beyond images such as texts and graphs. Different embedding and classification DNN models are required, yet the procedures of generating and merging seed regions and expanding training data may remain relatively intact.

REFERENCES

[1] B. Frey and N. Jojic, "Transformation-invariant clustering using the EM algorithm," *IEEE TPAMI*, vol. 25, no. 1, pp. 1–17, 2003.

[2] X. Liu *et al.*, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[3] A. Som *et al.*, "Perturbation robust representations of topological persistence diagrams," in *ECCV*, Sep. 2018, pp. 617–635.

[4] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *33rd ICML*, vol. 48. PMLR, Jun. 2016, pp. 478–487.

[5] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE TPAMI*, vol. 35, no. 8, pp. 1798–1828, 2013.

[6] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *IEEE ICCV*, Dec. 2015, pp. 1422–1430.

[7] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, Oct. 2016, pp. 69–84.

[8] C. Vondrick *et al.*, "Tracking emerges by colorizing videos," in *ECCV*, Sep. 2018, pp. 391–408.

[9] A. Vedaldi, Y. Asano, and C. Rupprecht, "Self-labelling via simultaneous clustering and representation learning," in *ICLR*, 2020, pp. 1–22.

[10] W. Van Gansbeke *et al.*, "Scan: Learning to classify images without labels," in *ECCV*, 2020, pp. 268–285.

[11] S. Park *et al.*, "Improving unsupervised image clustering with robust learning," in *IEEE CVPR*, Jun. 2021, pp. 12 278–12 287.

[12] S. Tammina, "Transfer learning using VGG-16 with deep convolutional neural network for classifying images," *IJSRP*, vol. 9, no. 10, pp. 143–150, 2019.

[13] J. Devlin *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019, pp. 4171–4186.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[16] J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database," in *IEEE CVPR*, Jun. 2009, pp. 248–255.

[17] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS*, Jan. 2001, pp. 849–856.

[18] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[19] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *IEEE ICCV*, Oct. 2019, pp. 9865–9874.

[20] F. Zhuang *et al.*, "A comprehensive survey on transfer learning," *IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.