Associative Deep Clustering: Training a Classification Network with no Labels

Philip Haeusser Johannes Plapp Vladimir Golkov Elie Aljalbout
Daniel Cremers
Dept. of Informatics, TU Munich
{haeusser, plapp, golkov, aljalbout, cremers}@in.tum.de

Abstract

Humans are able to look at a large number of images, find similarities and group images together by an abstract understanding of the content. Researchers have been trying to implement such unsupervised learning schemes for a long time: Given a dataset, e.g. images, find a rule to assign each example to one of k clusters. We propose a novel training schedule for neural networks that facilitates fully unsupervised end-to-end clustering that is direct, i.e. outputs a probability distribution over cluster memberships.

A neural network maps images to embeddings. We introduce centroid variables that have the same shape as image embeddings. These variables are jointly trained with the network's parameters. This is achieved by a cost function that associates the centroid variables with the embeddings of input images. Finally, an additional layer maps embeddings to logits allowing for the direct estimation of the respective cluster membership. Unlike other methods, this does not require any additional classifier to be trained on the embeddings in a separate step.

The proposed approach achieves state-of-the-art results in unsupervised classification and we provide an extensive ablation study to demonstrate its capabilities.

1. Introduction

1.1. Towards direct deep clustering

Deep neural networks have shown impressive potential on a multitude of computer vision challenges [38, 9, 39, 37, 7, 29]. A fundamental limitation in many applications is that they traditionally require huge amounts of labeled training data. To circumvent this problem, a plethora of semi-supervised and unsupervised training schemes have been proposed [6, 8, 25, 19]. They all aim at reducing the number of labeled data while leveraging large quantities of unlabeled data.

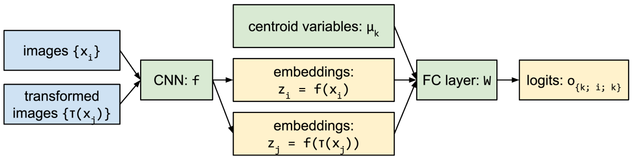

Figure 1: Associative deep clustering. Images (x_i) and transformations of them ($\tau(x_j)$) are sent through a CNN in order to obtain embeddings z. We introduce k centroid variables μ_k that have the same dimensionality as the embeddings. Our proposed loss function simultaneously trains these centroids and the network's parameters along with a mapping from embedding space to a cluster membership distribution.

It is an intriguing idea to train a neural network without any labeled data at all by automatically discovering structure in data given as minimal prior knowledge the number of classes. In many real-world applications beyond the scope of academic research, it is desired to discover structures in large unlabeled data sets. When the goal is to separate date into groups, this maneuver is called *clustering*. Deep neural networks are the model of choice when it comes to image understanding. In the past, however, deep neural networks have rarely been trained for clustering directly. A more common approach is *feature learning*. Here, a proxy task is designed to generate a loss signal that can be used to update the model parameters in an entirely unsupervised manner. An exhaustive comparison of previous works is collected in Section 1.2. All these approaches aim at transforming an input image x_i to a representation or embedding z_i that allows for clustering the data. In order to perform this last step, a mapping from embedding space to clusters is necessary, e.g. by training an additional classifier on the features, such as k-means [28] or an SVM [36] in a separate step.

A common problem in unsupervised learning is that there is no signal that tells the network to cluster different examples from the same class together although they look very different in pixel space. We call this the *blue sky problem* with the pictures of a flying bird and a flying airplane in mind, both of which will contain many blue pixels but just a few others that make the actual distinction. In particular auto-encoder approaches suffer from the blue sky problem as their cost function usually penalizes reconstruction in pixel space and hence favors the encoding of potentially unnecessary information such as the color of the sky.

1.2. Related work

Classical clustering approaches are limited to the original data space and are thus not very effective in high-dimensional spaces with complicated data distributions, such as images. Early deep-learning-based clustering methods first train a feature extractor, and in a separate step apply a clustering algorithm to the features [10, 32, 34].

Well-clusterable deep representations are characterized by high within-cluster similarities of points compared to low across-cluster similarities. Some loss functions optimize only one of these two aspects. Particularly, methods that do not encourage across-cluster dissimilarities are at risk of producing worse (or theoretically even trivial) representations/results [42], but some of them nonetheless work well in practice. Other methods avoid trivial representations by using additional techniques unrelated to clustering, such as autoencoder reconstruction loss during pre-training or entire training [20, 41, 42, 26].

Deep Embedded Clustering (DEC) [41] model simultaneously learns feature representations and cluster assignments. To get a good initialization, DEC needs autoencoder pretraining. Also, the approach has some issues scaling to larger datasets such as STL-10.

Variational Deep Embedding (VaDE) [44] is a generative clustering approach based on variational autoencoders. It achieves significantly more accurate results on small datasets, but does not scale to larger, higher-resolution datasets. For STL-10, it uses a network pretrained on the dramatically larger Imagenet dataset.

Joint Unsupervised Learning (JULE) of representations and clusters [43] is based on agglomerative clustering. In contrast to our method, JULE's network training procedure alternates between cluster updates and network training. JULE achieves very good results on several datasets. However, its computational and memory requirements are relatively high as indicated by [17].

Clustering convolutional neural networks (CCNN) [17] predict clustering assignments at the last layer, and a clustering-friendly representation at an intermediate layer. The training loss is the difference between the clustering predictions and the results of k-means applied to the learned representation. Interestingly, CCNN yields good results in practice, despite both the clustering features and the cluster predictions being initialized in random and contradictory ways. CCNN requires running k-means in each iteration.

Deep Embedded Regularized Clustering (DEPICT) [5] is similar to DEC in that feature representations and cluster assignments are simultaneously learned using a deep network. An additional regularization term is used to balance the cluster assignment probabilities allowing to get rid of the pretraining step. This method achieved a performance comparable to ours on MNIST and FGRC. However, it requires pre-training using the autoencoder reconstruction loss.

In Categorical Generative Adversarial Networks (CatGANs) [35], unlike standard GANs, the discriminator learns to separate the data into k categories instead of learning a binary discriminative function. Results on large datasets are not reported. Another approach based on GANs is [31].

Information-Maximizing Self-Augmented Training (IMSAT) [18] is based on Regularized Information Maximization (RIM) [22], which learns a probabilistic classifier that maximizes the mutual information between the input and the class assignment. In IMSAT this mutual information is represented by the difference between the marginal and conditional distribution of those values. IMSAT also introduces regularization via self-augmentation. This is achieved via an additional term to the cost function which assures that the generated data point and the original are similarly assigned. For large datasets, IMSAT uses fixed pre-trained network layers.

Several other methods with various quality of results exist [3, 27, 40, 2, 33, 15].

In summary, previous methods either do not scale well to large datasets [41, 44, 35], have high computational and/or memory cost [43], require a clustering algorithm to be run during or after the training (most methods, e.g. [43, 17]), are at risk of producing trivial representations, and/or require some labeled data [18] or clustering-unrelated losses (for example additional autoencoder loss [20, 41, 42, 26, 44, 5, 1]) to (pre-)train (parts of) the network. In particular reconstruction losses tend to overestimate the importance of low level features such as colors.

In order to solve these problems, it is desirable to develop new training schemes that tackle the clustering problem as a *direct* end-to-end training of a neural network.

1.3. Contribution

In this paper, we propose *Associative Deep Clustering* as an end-to-end framework that allows to train a neural network directly for clustering. In particular, we introduce *centroid embeddings*: variables that look like embeddings of images but are actually part of the model. They can be optimized and they are used to learn a projection from embedding space to the desired dimensionality, e.g. logits $\in \mathbb{R}^k$ where k is the number of classes. The intuition is that the centroid variables carry over high-level information about the data structure (i.e. cluster centroid embeddings) from iteration to iteration. It makes sense to train a neural network directly for a clustering task, rather than a proxy task (such as reconstruction from embeddings) since the ultimate goal is actually clustering.

To facilitate this, we introduce a cost function consisting of multiple terms which cause clusters to separate while associating similar images. *Associations* [13] are made between centroids and image embeddings, and between embeddings of images and their transformations. Unlike previous methods, we use clustering-specific loss terms that allow to directly learn the assignment of an image to a cluster. There is no need for a subsequent training procedure on the embeddings. The output of the network is a cluster membership distribution for a given input image. We demonstrate that this approach is useful for clustering images without any prior knowledge other than the number of classes.

The resulting learned cluster assignments are so good that subsequently re-running a clustering algorithm on the learned embeddings does not further improve the results (unlike e.g. [43]).

To the best of our knowledge, we are the first to introduce a training scheme for direct clustering jointly with network training, as opposed to feature learning approaches where the obtained features need to be clustered by a second algorithm such as k-means (as in most methods), or to methods where the clustering is directly learned but parts of the network are pre-trained and fixed [18].

Moreover, unlike most methods, we use *only* clustering-specific and invariance-imposing losses and no clustering-unrelated losses such as autoencoder reconstruction or classification-based pre-training.

In summary, our contributions are:

- We introduce centroid variables that are jointly trained with the network's weights.
- This is facilitated by our clustering cost function that makes *associations* between cluster centroids and image embeddings. No labels are needed at any time. In particular, there is no subsequent clustering step necessary such as k-means.
- We conducted an extensive ablation study demonstrating the effects of our proposed training schedule.
- Our method outperforms the current state of the art on a number of datasets.
- All code is available as an open-source implementation in TensorFlow[1].

[1] The code will be made available upon publication of this paper.

2. Associative Deep Clustering

In this section, we describe our setup and the cost function. Figure 2 depicts an overall schematic which will be referenced in the following.

2.1. Associative learning

Recent works have shown that *associations* in embedding space can be used for semi-supervised training and domain adaptation [13, 12]. Both applications require an amount of labeled training data which is fed through a neural network along with unlabeled data. Then, an imaginary walker is sent from embeddings of labeled examples to embeddings of unlabeled examples and back. From this idea, a cost function is constructed that encourages consistent association cycles, meaning that two labeled examples are associated via an unlabeled example with a high probability if the labels match and with a low probability otherwise. More formally: Let A_i and B_j be embeddings of labeled and unlabeled data, respectively. Then a similarity matrix $M_{ij} := A_i \cdot B_j$ can be defined. These similarities can now be transformed into a transition probability matrix by softmaxing M over columns:

$$\begin{aligned} P_{ij}^{ab} = P(B_j|A_i) :=& (\text{softmax}_{\text{cols}}(M))_{ij} \\ =& \exp(M_{ij}) / \sum_{j'} \exp(M_{ij'}) \end{aligned} \tag{1}$$

Analogously, the transition probabilities in the other direction (P^{ba}) are obtained by replacing M with M^T. Finally, the metric of interest is the probability of an association cycle from A to B to A:

$$\begin{aligned} P_{ij}^{aba} :=& (P^{ab}P^{ba})_{ij} \\ =& \sum_k P_{ik}^{ab} P_{kj}^{ba} \end{aligned} \tag{2}$$

Since the labels of A are known, it is possible to define a target distribution where inconsistent association cycles (where labels mismatch) have zero probability:

$$T_{ij} := \begin{cases} 1/\#\text{class}(A_i) & \text{class}(A_i) = \text{class}(A_j) \\ 0 & \text{else} \end{cases} \tag{3}$$

Now, the associative loss function becomes $\mathcal{L}_{\text{assoc}}(A, B) := \text{crossentropy}(T_{ij}; P_{ij}^{ab})$. For more details, the reader is kindly referred to [13].

2.2. Clustering loss function

In this work, we further develop this setup since there is no labeled batch A. In its stead, we introduce centroid variables μ_k that have the same dimensionality as the embeddings and that have surrogate labels $0, 1, \ldots, k-1$. With them and embeddings of (augmented) images, we can define clustering associations.

We define two associative loss terms:

- $\mathcal{L}_{\text{assoc,c}} := \mathcal{L}_{\text{assoc}}(\mu_k; z_i)$ where associations are made between (labeled) centroid variables μ_k (instead of A) and (unlabeled) image embeddings z_i
- $\mathcal{L}_{\text{assoc,aug}} := \mathcal{L}_{\text{assoc}}(z_i; f(\tau(x_j)))$ where we apply 4 random transformations τ to each image x_j resulting in 4 "augmented" embeddings z_j that share the same surrogate label. The "labeled" batch A then consists of all augmented images, the "unlabeled" batch B of embeddings of the unaugmented images x_i.

For simplicity, we imply a sum over all examples x_i and x_j in the batch. The loss is then divided by the number of summands.

The transformation τ randomly applies cropping and flipping, Gaussian noise, small rotations and changes in brightness, saturation and hue.

All embeddings and centroids are regularized with an L2 loss $\mathcal{L}_{\text{norm}}$ to have norm 1. We chose this value empirically to avoid too small numbers in the 128-dimensional embedding vectors.

A fully-connected layer W projects embeddings (μ_k, z_i and z_j) to logits $o_{k,i,j} \in \mathbb{R}^k$. After a softmax operation, each logit vector entry can be interpreted as the probability of membership in the respective cluster. W is optimized through a standard cross-entropy classification loss $\mathcal{L}_{\text{classification}}$ where the inputs are μ_k and the desired outputs are the surrogate labels of the centroids.

Finally, we define a transformation loss

$$\mathcal{L}_{\text{trafo}} := |1 - f(x_i)^T f(\tau(x_j)) - \text{crossentropy}(o_i, o_j)| \tag{4}$$

Here, we apply τ once and set it once to the identity such that the "augmented" batch contains each image and a transformed version of it. This formulation can be interpreted as a trick to obtain weak labels for examples since the logits yield an estimate for the class membership "to the best knowledge of the network so far". Particularly, this loss serves multiple purposes:

- The logit o_i of an image x_i and the logit o_j of the transformed version $\tau(x_j)$ should be similar for $i = j$.
- Images that the network thinks belong to the same cluster (i.e. their centroid distribution o is similar) should also have similar embeddings, regardless of whether τ was applied.
- Embeddings of one image and a different image are allowed to be similar if the centroid membership is similar.
- Embeddings are forced to be dissimilar if they do not belong to the same cluster.

The final cost function now becomes

$$\mathcal{L} = \alpha\mathcal{L}_{\text{assoc,c}} \tag{5}$$
$$+ \beta\mathcal{L}_{\text{assoc,aug}} \tag{6}$$
$$+ \gamma\mathcal{L}_{\text{norm}} \tag{7}$$
$$+ \delta\mathcal{L}_{\text{trafo}} \tag{8}$$

with the loss weights $\alpha, \beta, \gamma, \delta$.

In the remainder of this paper, we present an ablation study to demonstrate that the loss terms do in fact support the clustering performance. From this study, we conclude on a set of hyper parameters to be held fixed for all datasets. With these fixed parameters, we finally report clustering scores on various datasets along with a qualitative analysis of the clustering results.

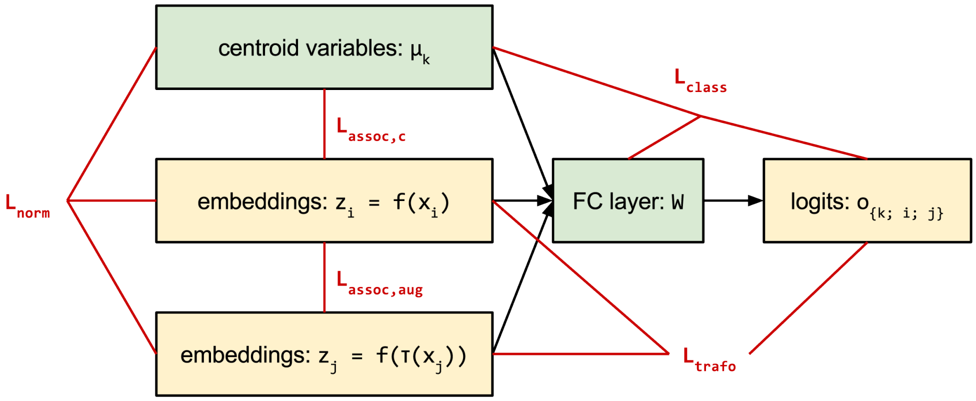

Figure 2: Schematic of the framework and the losses. Green boxes are trainable variables. Yellow boxes are representations of the input. Red lines connect the losses with their inputs. Please find the definitions in Section 2.2

	MNIST	FRGC	SVHN	CIFAR-10	STL-10
k-means on pixels	53.49	24.3	12.5	20.8	22.0
DEC [41]	84.30	37.8	-	-	35.9‡†
pretrain+DEC [18]	-	-	11.9 (0.4)†	46.9 (0.9)†	78.1 (0.1)†
VaDE [44]	94.46	-	-	-	84.5†
CatGAN [35]	95.73	-	-	-	-
JULE [43]	96.4	46.1	-	63.5‡‡	-
DEPICT [5]	96.5	**47.0**	-	-	-
IMSAT [18]	**98.4 (0.4)**	-	57.3 (3.9) ‡	45.6 (2.9)†	94.1†
CCNN [17]	91.6	-	-	-	-
ours (VCNN): Mean	**98.7 (0.6)**	43.7 (1.9)	37.2 (4.6)	26.7 (2.0)	38.9 (5.9)
ours (VCNN): Best	99.2	46.0	43.4	28.7	41.5
ours (ResNet): Mean	95.4 (2.9)	21.9 (4.9)	**38.6 (4.1)**	**29.3 (1.5)**	**47.8 (2.7)**
ours (ResNet): Best	97.3	29.0	**45.3**	**32.5**	**53.0**
ours (ResNet): K-Means	93.8 (4.5)	24.0 (3.0)	35.4 (3.8)	30.0 (1.8)	47.1 (3.2)

Table 1: Clustering. Accuracy (%) on the test sets (higher is better). Standard deviation in parentheses where available. We report the mean and best of 20 runs for two architectures. For ResNet, we also report the accuracy when k-means is run on top of the obtained embeddings after convergence. †: Using features after pre-training on Imagenet. ‡: Using GIST features. ‡†: Using Histogram-of-oriented-gradients (HOG) features. ‡‡: Using 5k labels to train a classifier on top of the features from clustering.

3. Experiments

3.1. Training procedure

For all experiments, we start with a warm-up phase where we set all loss weights to zero except $\beta = 0.9$ and $\gamma = 10^{-5}$. $\mathcal{L}_{\text{assoc,aug}}$ is used to initialize the weights of the network. After 5,000 steps, we find an initialization for μ_k by running k-means on the training embeddings. Then, we activate the other loss weights.

We use the Adam optimizer [21] with (beta1=0.8, beta2=0.9) and a learning rate of 0.0008. This learning rate is divided by 3 every 10,000 iterations. Following [11] we also adopt a warmup-phase of 2,000 steps for the learning rate.

We report results on two architectures: A vanilla convolutional neural net from [14] (in the following referred to as VCNN) and the commonly used ResNet architecture [16]. For VCNN, we adopt the hyper-parameters used in the original work, specifically a mini-batch size of 100 and embedding size 128. For ResNet, we use the architecture specified for CIFAR-10 by [16] for all datasets except STL-10. Due to the larger resolution of this dataset, we adopt the ImageNet variant, and modify it in the following way:

- The kernel size for the first convolutional layer becomes 3, with a stride of 1.
- The subsequent max-pooling layer is set to 2x2
- We reduce the number of filters by a factor of 2.

We use a mini-batch size of 128 images. For ResNet, we use 64-dimensional embedding vectors, as the CIFAR10-variant only has 64 filters in the last layer. We use a block size of 3 for MNIST, FRGC and STL-10, and 5 for SVHN and CIFAR-10.

The visit weight for both association losses is set to 0.3.

In order to find reasonable loss weights and investigate the importance, we conducted an ablation study which is described in the following section.

3.2. Ablation study

We randomly sampled the loss weights for all previously introduced losses in the range [0; 1] except for the normalization loss weight γ and ran 1,061 experiments on MNIST. Then, we used this clustering model to assign classes to the MNIST test set. Following [41], we picked the permutation of class labels that reflects the true class labels best and summarized the respective accuracies in Figure 5. For each point in a plot, we held only the one parameter fixed and averaged all according runs. This explains the error bars which arise from variance of all other parameters. It can still be seen very clearly that each loss has an important contribution to the test accuracy indicating the importance of the respective terms in our cost function.

We carried out similar experiments for other datasets which allowed to choose one set of hyper parameters for all subsequent experiments, which is described in the next section.

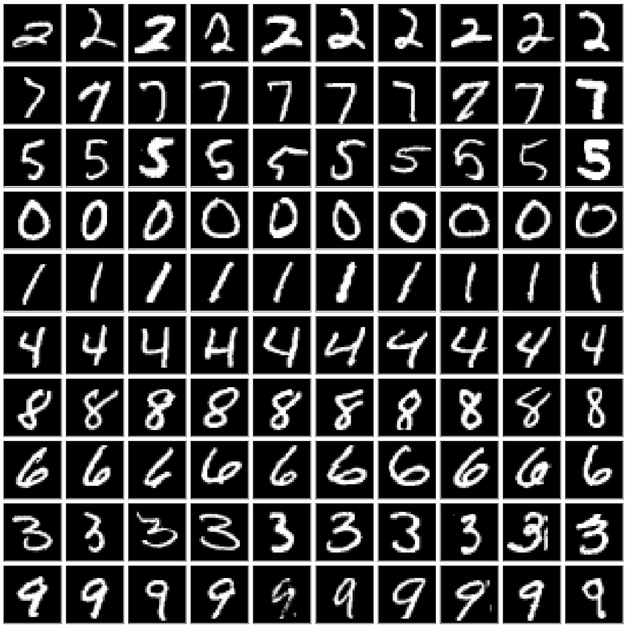

Figure 3: Clustering examples from MNIST. Each row contains the examples with the highest probability to belong to the respective cluster.

3.3. Clustering performance

From the previous section, we chose the following hyper parameters to hold fixed for all datasets:
$\alpha = 1; \beta = 0.9; \gamma = 10^{-5}; \delta = 2 \times 10^{-5}$

3.4. Evaluation protocol

Following [41], we set k to be the number of ground-truth classes in each dataset, and evaluate the clustering performance using the unsupervised clustering accuracy (ACC) metric. For every dataset, we run our proposed algorithm 10 times, and report multiple statistics:

- Mean and standard deviation of all clustering accuracies.
- The maximum clustering accuracy of all runs.

3.5. Datasets

We evaluate our algorithm on the following, widely-used image datasets:

MNIST [24] is a benchmark containing 70,000 hand-written digits. We use all images from the training set without their labels. Following [13], we set the visit weight to 0.8. We also use 1,000 warm-up steps.

For **FRGC**, we follow the protocol introduced in [43], and use the 20 selected subsets, providing a total of 2,462 face images. They have been cropped to 32×32px images. We use a visit weight of 0.1 and 1,000 warm-up steps.

SVHN [30] contains digits extracted from house numbers in Google Street View images. We use the training set combined with 150,000 images from the additional, unlabeled set for training.

CIFAR-10 [23] contains tiny images of ten different object classes. Here, we use all images of the training set.

STL-10 [4] is similar to CIFAR-10 in that it also has 10 object classes, but at a significantly higher resolution (96×96px). We use all 5,000 images of the training set for our algorithm. We do not use the unlabeled set, as it contains images of objects of other classes that are not in the training set. We randomly crop images to 64×64px during training, and use a center crop of this size for evaluation.

Table 1 summarizes the results. We achieve state-of-the-art results on MNIST, SVHN, CIFAR-10 and STL-10.

3.6. Qualitative analysis

Figure 3 and Figure 4 show images from the MNIST and STL-10 test set, respectively. The examples shown have the highest probability to belong to the respective clusters (logit argmax). The MNIST examples reveal that the network has learned to cluster hand-written digits well while generalizing to different ways how digits can be written. For example, *2* can be written with a little loop. Some examples of *8* have a closed loop, some don't. The network does not make a difference here which shows that the proposed training scheme does learn a high-level abstraction.

Analogously, for STL-10, nearly all images are clustered correctly, despite a broad variance in colors and shapes. It is interesting to see that there is no bird among the top-scoring samples of the *airplane*-cluster, and all birds are correctly clustered even if they are in front of a blue background. This demonstrates that our proposed algorithm does not solely rely on low-level features such as color (cf. the *blue sky problem*) but actually finds common patterns based on more complex similarities.

4. Conclusion

We have introduced *Associative Deep Clustering* as a novel, direct clustering algorithm for deep neural networks. The central idea is to jointly train centroid variables with the network's weights by using a clustering cost function. No labels are needed at any time and our approach does not require subsequent clustering such as many feature learning schemes. The importance of the loss terms were demonstrated in an ablation study and the effectiveness of the training schedule is reflected in state-of-the-art results in classification. A qualitative investigation suggests that our method is able to successfully discover structure in image data even when there is high intra-class variation. In clustering, there is no absolute right or wrong - multiple solutions can be valid, depending on the categories that a human

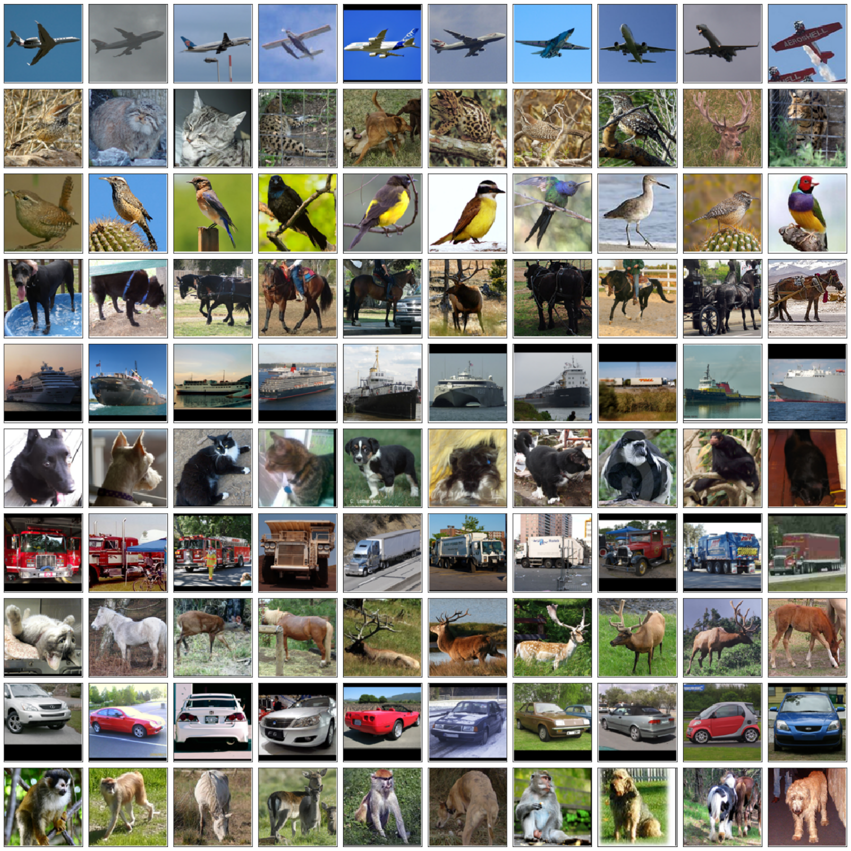

Figure 4: Clustering examples from STL-10. Each row contains the examples with the highest probability to belong to the respective cluster.

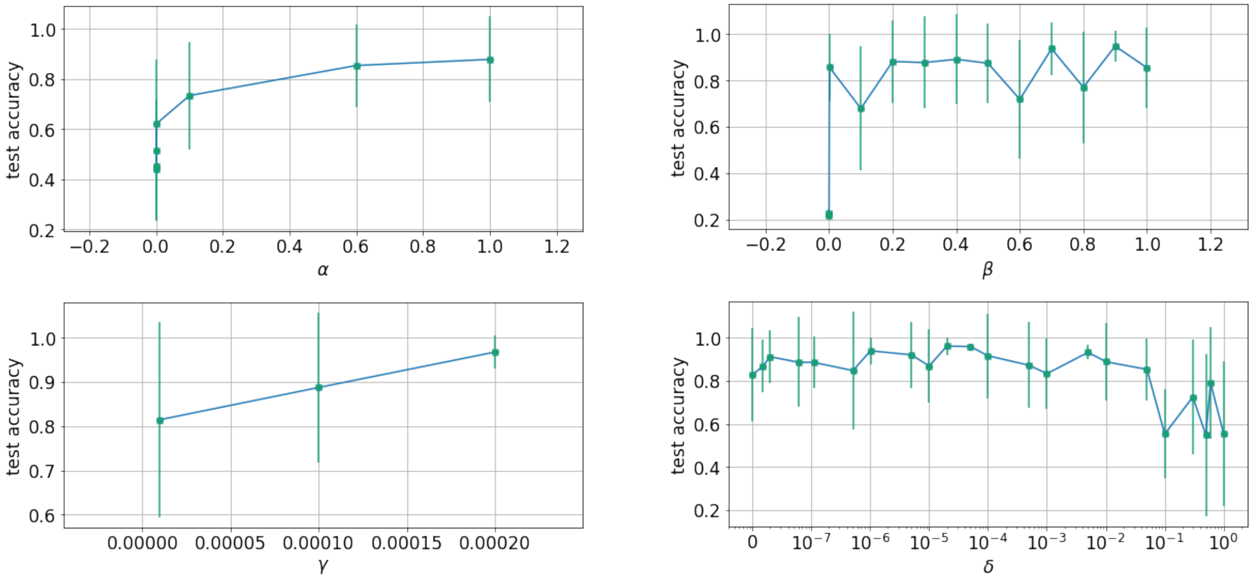

Figure 5: Ablation study for different hyper-parameters on MNIST.

introduces. We believe, however, that our formulation is applicable to many real world problems and that the simple implementation will hopefully inspire many future works.

References

[1] E. Aljalbout, V. Golkov, Y. Siddiqui, and D. Cremers. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*, 2018. 2

[2] D. Chen, J. Lv, and Z. Yi. Unsupervised multi-manifold clustering by learning deep representation. In *Workshops at the 31th AAAI conference on artificial intelligence (AAAI)*, pages 385–391, 2017. 2

[3] G. Chen. Deep learning with nonparametric clustering. *arXiv preprint arXiv:1501.03084*, 2015. 2

[4] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 6

[5] K. G. Dizaji, A. Herandi, and H. Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. *arXiv preprint arXiv:1704.06327*, 2017. 2, 5

[6] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 1

[7] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015. 1

[8] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 766–774, 2014. 1

[9] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014. 1

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 2

[11] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 5

[12] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 3

[13] P. Haeusser, A. Mordvintsev, and D. Cremers. Learning by association - a versatile semi-supervised training method for neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3, 6

[14] P. Haeusser, A. Mordvintsev, and D. Cremers. Learning by association - a versatile semi-supervised training method for neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[15] W. Harchaoui, P.-A. Mattei, and C. Bouveyron. Deep adversarial gaussian mixture auto-encoder for clustering. 2017. 2

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[17] C.-C. Hsu and C.-W. Lin. CNN-based joint clustering and representation learning with feature drift compensation for large-scale image data. *arXiv preprint arXiv:1705.07091*, 2017. 2, 5

[18] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama. Learning discrete representations via information maximizing self augmented training. *arXiv preprint arXiv:1702.08720*, 2017. 2, 3, 5

[19] C. Huang, C. Change Loy, and X. Tang. Unsupervised learning of discriminative attributes and visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5175–5184, 2016. 1

[20] P. Huang, Y. Huang, W. Wang, and L. Wang. Deep embedding network for clustering. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 1532–1537. IEEE, 2014. 2

[21] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[22] A. Krause, P. Perona, and R. G. Gomes. Discriminative clustering by regularized information maximization. In *Advances in neural information processing systems*, pages 775–783, 2010. 2

[23] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 6

[24] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 6

[25] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013. 1

[26] F. Li, H. Qiao, B. Zhang, and X. Xi. Discriminatively boosted image clustering with fully convolutional auto-encoders. *arXiv preprint arXiv:1703.07980*, 2017. 2

[27] Y. Lukic, C. Vogt, O. Dürr, and T. Stadelmann. Speaker identification and clustering using convolutional neural networks. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016. 2

[28] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967. 1

[29] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. 1

[30] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011. 6

[31] V. Premachandran and A. L. Yuille. Unsupervised learning using generative adversarial training and clustering. 2016. 2

[32] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[33] S. Saito and R. T. Tan. Neural clustering: Concatenating layers for better projections. 2017. 2

[34] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016. 2

[35] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015. 2, 5

[36] J. A. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999. 1

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 1

[38] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *Advances in Neural Information Processing Systems*, pages 2553–2561, 2013. 1

[39] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014. 1

[40] Z. Wang, S. Chang, J. Zhou, M. Wang, and T. S. Huang. Learning a task-specific deep architecture for clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 369–377. SIAM, 2016. 2

[41] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, pages 478–487, 2016. 2, 5, 6

[42] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. *arXiv preprint arXiv:1610.04794*, 2016. 2

[43] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5147–5156, 2016. 2, 3, 5, 6

[44] Y. Zheng, H. Tan, B. Tang, H. Zhou, et al. Variational deep embedding: A generative approach to clustering. *arXiv preprint arXiv:1611.05148*, 2016. 2, 5