SUBSPACE CLUSTERING VIA ROBUST SELF-SUPERVISED CONVOLUTIONAL NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep subspace clustering (SC) algorithms recently gained attention due to their ability to successfully handle nonlinearities in data. However, the insufficient capability of existing SC methods to deal with data corruption of unknown (arbitrary) origin hinders their generalization ability and applicability to address real-world data clustering problems. This paper proposes the robust formulation of the self-supervised convolutional subspace clustering network (S^2ConvSCN) that incorporates the fully connected (FC) layer and, with an additional spectral clustering module, is capable of estimating the clustering error without using the ground truth. Robustness to data corruptions is achieved by using the correntropy induced metric (CIM) of the error that also enhanced the generalization capability of the network. The experimental finding showed that CIM reduces sensitivity to overfitting during the learning process and yields better clustering results. In a truly unsupervised training environment, Robust S^2ConvSCN outperforms its baseline version by a significant amount for both seen and unseen data on four well-known datasets.

1 INTRODUCTION

Subspace clustering approaches have achieved encouraging performance when compared with the clustering algorithms that rely on proximity measures between data points. The main idea behind the subspace model is that the data can be drawn from low-dimensional subspaces which are embedded in a high-dimensional ambient space (Lodhi & Bajwa, 2018). Grouping such data associated with respective subspaces is known as the subspace clustering problem (Vidal, 2011). That is, each low-dimensional subspace corresponds to a class or category. Up to now, two main approaches for recovering low-dimensional subspaces are developed: models that are based on the self-representation property, and non-linear generalization of subspace clustering called *union of subspaces* (UoS) (Lodhi & Bajwa, 2018; Lu & Do, 2008; Wu & Bajwa, 2014; 2015b). UoS algorithms are out of the scope of this work. Self-representation subspace clustering is achieved in two steps: (i) learning representation matrix $\mathbf{C}$ from data $\mathbf{X}$ and building corresponding affinity matrix $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$; (ii) clustering the data into k clusters by grouping the eigenvectors of the graph Laplacian matrix that correspond with the leading k eigenvalues. This second step is known as spectral clustering (Ng et al., 2002; Von Luxburg, 2007). Owning to the presumed subspace structure, the data points obey the self-expressiveness or self-representation property (Elhamifar & Vidal, 2013; Peng et al., 2016b; Liu et al., 2012; Li & Vidal, 2016; Favaro et al., 2011). In other words, each data point can be represented as a linear combination of other points in a dataset: **X=XC**.

The self-representation approach is facing serious limitations regarding real-world datasets. One limitation relates to the linearity assumption because in a wide range of applications samples lie in nonlinear subspaces, e.g. face images acquired under non-uniform illumination and different poses (Ji et al., 2017). Standard practice for handling data from nonlinear manifolds is to use the kernel trick on samples mapped implicitly into high dimensional space. Therein, samples better conform to linear subspaces (Patel et al., 2013; Patel & Vidal, 2014; Xiao et al., 2015; Brbić & Kopriva, 2018). However, identifying an appropriate kernel function for a given data set is quite a difficult task (Zhang et al., 2019b). The second limitation of existing deep SC methods relates to their assumption that the origin of data corruption is known, in which case the proper error model can be employed. In real-word applications origin of data corruption is unknown. That can severely harm the algorithm's learning process if the non-robust loss function is used. Furthermore, validation

(i.e. stopping of the learning process) in most of the deep SC methods often requires access to the ground-truth labels. That stands for violation of the basic principle of unsupervised machine learning and yields the overly-optimistic results. Dataset size is also a limitation when it comes to memory requirements. Since the self-representation subspace clustering is based on building the affinity matrix, memory complexity increases as the square of the dataset size. However, the latter limitation is not in the main focus of this work.

Motivated by the exceptional ability of deep neural networks to capture complex underlying structures of data and learn discriminative features for clustering (Hinton & Salakhutdinov, 2006; Dilokthanakul et al., 2016; Ghasedi Dizaji et al., 2017; Tian et al., 2014; Xie et al., 2016), deep subspace clustering approaches emerged recently (Ji et al., 2017; Abavisani & Patel, 2018; Peng et al., 2016a; Yang et al., 2019; Zhou et al., 2018; Ji et al., 2019b; Peng et al., 2018; 2017; Zhou et al., 2019; Zhang et al., 2019a; Kheirandishfard et al., 2020). In particular, it is shown that convolutional neural networks (CNNs), when applied to images of different classes, can learn features that lie in a UoS (Lezama et al., 2018). Mostly, the base of the recently developed deep subspace-clustering networks is convolutional autoencoder. It is an end-to-end fully convolutional network that is based on the minimization of the reconstruction error. Together, the autoencoder and an additional self-expression (SE) module are forming a Deep subspace clustering network (DSCNet) (Ji et al., 2017). Hence, the total loss function of DSCNet is composed of reconstruction loss and SE model loss. That is, during the learning process the clustering quality is not taken into account. Self-supervised convolutional SC network (S^2ConvSCN) (Zhang et al., 2019a) addressed this issue through the addition of a fully connected layer (FC) module and a spectral clustering module that, respectively, generate soft- and pseudo-labels. Dual self-supervision is achieved by forcing these two modules to converge towards consensus. Related accumulated loss, therefore, participates in enhancing the self-representation matrix and the quality of features extracted in the encoder layer. The architecture of S^2ConvSCN has a possibility of direct classification once the learning process is completed. A trained encoder and the FC module can make a new network that can directly classify unseen data, also known as an out-of-sample problem. However, while this network can be validated and compared with other algorithms on a separate data set, such an ablation study was not completed. Furthermore, the main disadvantage of the DSCNet architecture and indirectly S^2ConvSCN is that the network training is stopped when the accuracy is highest (Ji et al., 2019a). First, it is a direct violation of the unsupervised learning principle as the ground-truth labels are exposed. Second, the reported performance (Zhang et al., 2019a; Ji et al., 2017) is overly-optimistic and can not be compared to other algorithms.

Considering the mentioned disadvantages, basic research questions of this work are the following:

- Can the S^2ConvSCN architecture be more robust to random and sample-specific errors?
- Is there a self-stopping criterion that yields to more realistic training results?
- Can a fair performance estimation for deep subspace clustering models be established?

Mentioned research questions led to three main contributions of proposed Robust S^2ConvSCN: (i) robustness to errors of the unknown (arbitrary) origin is achieved by using the correntropy induced metric (CIM) in the self-expression loss; (ii) the network is trained using the early-stopping method while monitoring only the accumulated loss; (iii) thanks to correntropy based loss function the training process is less sensitive to data corruptions which enables the network to generalize better. This study has, also, two side-contributions: (i) the performance of models is estimated using the unseen (out-of-sample) data; (ii) block-diagonal regularization of self-representation matrix is integrated into the gradient descent learning process. A complete head to head comparison of the baseline S^2ConvSCN model and our robust approach can be seen in Figure 1.

2 Background and related work

2.1 Main notations and definitions

Throughout this paper, matrices are represented with bold capital symbols and vectors with bold lower-case symbols. $\mathbf{X} \in \mathbb{R}^{d \times N}$ represents data matrix comprised of N data samples with dimensionality d. $\left\{\mathbf{H}_i^{(l)}\right\}_{i=1}^{m^{(l)}}$ represent feature maps produced at the output of layer $l-1$. Thus, $\mathbf{H}^{(0)} = \mathbf{X}$

	Self-expressive layer	Early stopping	Evaluation	Robustness
Baseline approach:	$\mathrm{MSE} + \ell_2$	Highest accuracy	On in-sample labels used for hyperparameter optimization	To additive white Gaussian noise
Our approach:	$\mathrm{CIM} + \ell_2$ $\mathrm{CIM} + \mathrm{BD}$ $\frac{1}{2}\|\mathbf{C} - \mathbf{A}\|^2$	Loss decrease plateau	On out-of-sample labels previously unseen by the model	To data corruption of unknown (arbitrary) origin

Figure 1: The figure shows a comparison between the baseline S^2ConvSCN and our approach. Key differences illustrate the contributions of this work. The last regularization in the self-expressive layer stands for symmetric loss introduced in Equation (9).

and $\mathbf{H}^{(L)} = \hat{\mathbf{X}}$. $\hat{\mathbf{X}}$ represents the output of the decoder and L represents number of convolutional layers in the autoencoder. $\left\{\mathbf{w}_i^{(l)}\right\}_{i=1}^{m^{(l)}}$ stand for a set of filters with associated biases $\left\{\mathbf{b}_i^{(l)}\right\}_{i=1}^{m^{(l)}}$ that form a convolutional layer $l = 1, \ldots, L$. $\mathbf{z}_n = \left[\mathbf{h}_1^{(L/2)}(:) \ldots \mathbf{h}_{m^{(L/2)}}^{(L/2)}(:)\right]^T \in \mathbb{R}^{\hat{d} \times 1}$ stands for feature vector comprised of vectorized and concatenated feature maps, with $\hat{d}$ extracted features, in the top layer $\frac{L}{2}$ (encoder output) representing input sample $\mathbf{x}_n, n = 1, \ldots, N$. $\mathbf{C} \in \mathbb{R}^{N \times N}$ stands for representation matrix in self-expressive model $\mathbf{Z} = \mathbf{Z}\mathbf{C}$. $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}^T|$ is the affinity matrix and $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$ is corresponding graph Laplacian matrix. $\mathbf{D}$ is diagonal degree matrix such that $\mathbf{D}_{ii} = \sum_{j=1}^{N} \mathbf{A}_{ij}$. $\|\mathbf{X}\|_F = \sqrt{\sum_{i,j=1}^{N} x_{ij}^2}$ is the Frobenius norm of matrix $\mathbf{X}$. $\ell_p(\mathbf{x}) = \|\mathbf{x}\|_p = (\sum_{i=1}^{d} \|x_i\|^p)^{1/p},\ 0 < p \leq 1$ is the ℓ_p norm of $\mathbf{x}$. $\ell_0(\mathbf{x}) = \|\mathbf{x}\|_0 = \#\{x_i \neq 0,\ i = 1, \ldots, d\}$, where # denotes the cardinality function, is ℓ_0 quasi norm of $\mathbf{x}$. The S_p, $0 < p \leq 1$, Schatten norms of matrix $\mathbf{X}$ are defined as the corresponding ℓ_p norms of the vector of singular values of $\mathbf{X}$, i.e. $S_p(\mathbf{X}) = \|\sigma(\mathbf{X})\|_p$ where $\sigma(\mathbf{X})$ stands for the vector of singular values of $\mathbf{X}$. Depending on the context, $\mathbf{0}$ represents matrix/vector of all zeros and $\mathbf{1}$ represents matrix/vector of all ones.

Grouping the data according to the linear subspaces they are drawn from is known as subspace clustering (Vidal, 2011). The problem is formally defined in:

Definition 1. Let $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_k]$ be a set of sample vectors drawn from a union of k subspaces in $\mathbb{R}^d$, $\cup_{i=1}^{k} \{S_i\}$, of dimensions $d_i \ll \min\{d, N\}$, for $i = 1, \ldots, k$. Let $\mathbf{X}_i$ be a collection of N_i samples drawn from subspace S_i, $N = \sum_{i=1}^{k} N_i$. The problem of subspace clustering is to segment samples into the subspaces they are drawn from. Throughout this paper, as it is the case in the majority of other papers, we have assumed that number of clusters k is known *a priori*.

2.2 Approaches to Subspace clustering

Usually, processes that operate in different modes generate data in real-world scenarios. Each mode models such data as lying on a subspace, while the whole process, thus, generates data lying on a union of subspaces (UoS) (Lodhi & Bajwa, 2018). Such UoS approach has been introduced in (Lu & Do, 2008; Wu & Bajwa, 2014; 2015b;a) and inspired some studies in different frameworks such as compressive sensing (Joneidi et al., 2016; Yap & Pribić, 2014; Davenport et al., 2010), general UoS (Lodhi & Bajwa, 2018; Gini et al., 2004), and linear sampling of UoS (Wimalajeewa et al., 2015). The alternative to the UoS model is the self-representation based subspace model. It implies that every sample from the dataset can be represented as a linear combination of other samples from the same cluster. While shallow models directly optimize such a self-representation matrix, their deep counterparts train the whole network to better extract features from the raw data and achieve representation linearity.

Many approaches to deep subspace clustering are based on the introduction of the self-representation in the feature space (Abavisani & Patel, 2018; Ji et al., 2017; Peng et al., 2016a; Zhou et al., 2018; 2019; Zhang et al., 2019a; Kheirandishfard et al., 2020; Zhang et al., 2020). Thus, by the virtue of

this, improvements of the shallow subspace clustering methods are of direct relevance to their deep counterparts. The subspace clustering task is accomplished through (i) learning the representation matrix $\mathbf{C}$ from data $\mathbf{X}$, and (ii) clustering the data into k clusters by grouping the eigenvectors of the graph Laplacian matrix $\mathbf{L}$ that correspond with the k leading eigenvalues. This second step is known as spectral clustering (Ng et al., 2002; Von Luxburg, 2007). Low-rank (Liu et al., 2012; Favaro et al., 2011) and sparse models (Elhamifar & Vidal, 2013) are one of the commonly used algorithms to solve SC clustering problem. They aim to learn the low-rank and sparse representation matrix by solving the following optimization problem (Li & Vidal, 2016):

$$\min_{\mathbf{C}} \lambda \left\|\mathbf{C}\right\|_{S_p}^p + \tau \left\|\mathbf{C}\right\|_p^p \ s.t. \ \mathbf{Z} = \mathbf{ZC} + \mathbf{E}, \ diag(\mathbf{C}) = \mathbf{0} \tag{1}$$

where λ and τ are nonnegative regularization constants. If number of layers $L = 0$ problem (1) is related to shallow subspace clustering. Constraint $diag(\mathbf{C}) = \mathbf{0}$ is necessary to prevent sparseness regularized optimization algorithms to converge towards trivial solution where each data point represents itself. This constraint is not necessary for problem constrained only by low-rank. Low rank (Liu et al., 2012; Li & Vidal, 2016) and sparse (Elhamifar & Vidal, 2013) subspace clustering is obtained in (1) by setting $\tau = 0$ and $\lambda = 0$, respectively. When data samples are contaminated with additive white Gaussian noise (AWGN) problem (1) becomes:

$$\min_{\mathbf{C}} \left\|\mathbf{E}\right\|_F^2 + \lambda \left\|\mathbf{C}\right\|_{S_p}^p + \tau \left\|\mathbf{C}\right\|_p^p \ s.t. \ diag(\mathbf{C}) = \mathbf{0} \tag{2}$$

where **E=Z-ZC**. Alternatively, square of the Frobenius norm of $\mathbf{C}$ is used for regularization (Lu et al., 2012):

$$\min_{\mathbf{C}} \left\|\mathbf{E}\right\|_F^2 + \lambda \left\|\mathbf{C}\right\|_F^2 \tag{3}$$

Objective (3) is used also in the self-expression module of the S^2ConvSCN in (Zhang et al., 2019a). As seen from (2) and (3), the MSE measure for discrepancy between $\mathbf{Z}$ and its self-representation $\mathbf{ZC}$ is justified only for the contamination by the AWGN. For sample-specific corruptions (outliers) the proper norm is $\left\|\mathbf{E}\right\|_{2,1}$ while for large random corruptions the proper choice is $\left\|\mathbf{E}\right\|_1$ (Liu et al., 2012). However, errors in real world data have different origins and magnitude and may not follow specific probabilistic model. Sometimes, it is hard to know the true origin of corruptions present in data. Thus, to obtain method robust to arbitrary corruption we propose to introduce the CIM of the error. Rationale behind introduction of any regularization on $\mathbf{C}$ is to reflect its structural property of block-diagonality. Even though $\left\|\mathbf{C}\right\|_{S_p}$ and $\left\|\mathbf{C}\right\|_p$, $0 \leq p \leq 1$ in principle satisfy the enforced block-diagonality condition, their approximation of the BD structure of $\mathbf{C}$ is indirect (Lu et al., 2018). Hence, for comparison, this study proposes introduction of loss function with gradient-based BD regularization on representation matrix $\mathbf{C}$.

2.3 Baseline model description

The base of the DSCNet (Ji et al., 2017) is a fully convolutional autoencoder. Following the flattened latent code of the autoencoder is an additional self-expression module which forms the full architecture of the DSCNet. While it is producing representation matrix $\mathbf{C}$, the clustering algorithm can only cluster the in-sample data. Thus, clustering error is not part of the learning process and, therefore, does not influence the quality of learned features and representation matrix. According to the training procedure given in (Ji et al., 2019a), the performance of the DSCNet strongly depends on observing true labels every epoch. Although presented as an unsupervised method, DSCNet is using the minimum of clustering error as an early stopping criterion which is in contradiction with the principles of unsupervised learning and arguably leads to the overfitting and overly-optimistic performance estimation of the model. As can be seen in Figure 2, much better representation matrix $\mathbf{C}$ can be learned from a model by ignoring loss and stopping when the accuracy is highest. Another problematic part is that the DSCNet algorithm is a post-processing matrix $\mathbf{C}$ with a hyperparameter optimized using the ground truth labels. Furthermore, DSCNet architecture is a base for the S^2ConvSCN model (Zhang et al., 2019a), which is why S^2ConvSCN suffers from the majority of problems as DSCNet.

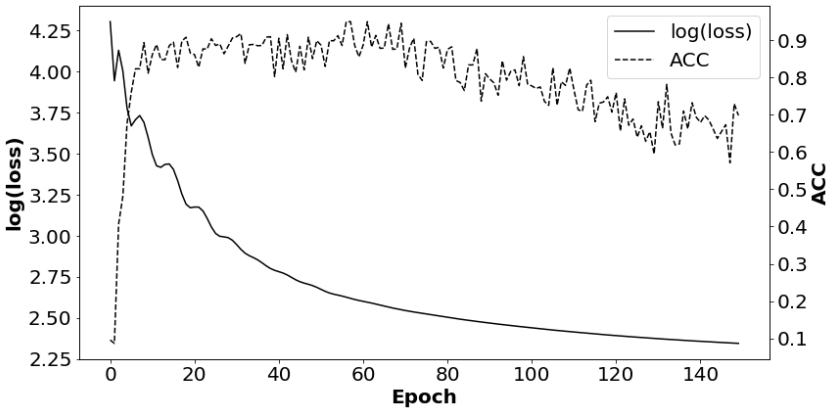

Figure 2: The figure illustrates the change of accuracy and loss over epochs for the original DSCNet model with MSE in the self-expressive layer. Model was trained and tested on a full COIL20 dataset with ℓ_2 regularization of representation matrix $\mathbf{C}$ (see Equation (5)). It can be seen that accuracy is oscillating even though the loss is decreasing. Thus, selecting the model requires access to ground truth labels and makes an unfair practice.

A novelty that is presented in S^2ConvSCN is an FC layer, attached to the latent code, that is upgrading the DSCNet model. On its output, FC has softmax which can directly classify the input data. Upon finishing the learning procedure, the encoder and FC layer form a new model that can be used for evaluating the learning procedure on the independent unseen set. While authors of the S^2ConvSCN model (Zhang et al., 2019a) provide a tool for dealing with the out-of-sample data, the actual performance on unseen data has not been tested. The second novelty of S^2ConvSCN is training the FC layer from pseudo-labels. When training the network, the spectral clustering module clusters the data according to the affinity matrix every E epochs and updates the pseudo-labels. It is important to note that the mentioned affinity matrix is constructed from the matrix $\mathbf{C}$ learned in the self-expression layer and it is changing every epoch during the training. Pseudo-labels assigned to the data are used in two ways. The first is to train the FC layer for soft classification, and the second is to suppress the $\mathbf{C}$ matrix values for samples that do not belong to the same cluster. Both ways are known as self-supervision. Keeping in mind that: (i) the S^2ConvSCN is upgraded DSCNet model, (ii) it is using the pretrained DSCNet parts of the network, and (iii) it has not been tested on the independent dataset, it is reasonable to conclude that the label-leakage also appeared in S^2ConvSCN. Regardless of having the self-supervision modules, the first pseudo-labels generated from DSCNet contain leaked knowledge about the group affiliations of the data.

Authors of both (Ji et al., 2017) and (Zhang et al., 2019a) suggest that to converge, parts of the model should be pretrained before training the whole model. However, there is no guaranty that the model will keep converging after attaching final layers or modules. For example, if a pretrained DSCNet yields reasonably good pseudo-labels, it is possible that after attaching the FC layer and performing self-supervision in S^2ConvSCN, the matrix $\mathbf{C}$ will get worse and the whole model will diverge from the optimal solution. Thus, it is important to tune constants associated with the loss functions of the mentioned modules. As the S^2ConvSCN has many loss functions regarding different layers or modules, a smaller learning rate could be beneficial in the sense that the overall loss can reach a better minimum in the error space. However, decreasing the learning rate on a plateau impacts the less-contributing losses. That can spoil the goodness of representation matrix $\mathbf{C}$ while the training process tries to reach the loss minimum.

3 Robust self-supervised convolutional subspace clustering network

Motivated by discussion in previous section, we propose two new objective functions $\mathcal{L}_{CIM}$ for the self-expression module of the S^2ConvSCN architecture:

$$\min_{\mathbf{C}} \mathrm{CIM}^2(\mathbf{E}) + \gamma \left\|\mathbf{C}\right\|_{[k]}, \tag{4}$$

$$\min_{\mathbf{C}} \mathrm{CIM}^2(\mathbf{E}) + \gamma \left\|\mathbf{C}\right\|_{2} \tag{5}$$

where $\mathbf{E} = \mathbf{Z} - \mathbf{ZC}$. $\mathbf{Z}$ and $\mathbf{ZC}$ stand for input and output features of self-expression layer, respectively. $\left\|\mathbf{C}\right\|_{[k]}$ denotes BD regularization and $\left\|\mathbf{C}\right\|_{2}$ denotes ℓ_2 regularization of representation matrix $\mathbf{C}$. γ represents a trade-off constant. The objectives (4) and (5) ensure a smooth decrease of the loss function that enables the use of label-free stopping criterion. For the sake of completeness, other loss functions of Robust S^2ConvSCN are stated. Auto-encoder reconstruction loss, where $\mathbf{X}$ represents the input data and $\hat{\mathbf{X}}$ represents the output of the decoder is defined as:

$$\mathcal{L}_{REC} = \frac{1}{2N}\sum_{j=1}^{N}\|\mathbf{x}_j - \hat{\mathbf{x}}_j\|_2^2 = \frac{1}{2N}\left\|\mathbf{X} - \hat{\mathbf{X}}\right\|_F^2 \tag{6}$$

When matrix $\mathbf{Q}$ is calculated, $\left\|\mathbf{C}\right\|_{\mathbf{Q}}$ loss is defined as:

$$\mathcal{L}_{CQ} = \sum_{i,j}|c_{ij}|\frac{\|\mathbf{q}_i - \mathbf{q}_j\|_2^2}{2} := \left\|\mathbf{C}\right\|_{\mathbf{Q}} \tag{7}$$

where $\mathbf{q}_i$ and $\mathbf{q}_j$ represent one-hot encoded pseudo-labels for i-th and j-th sample. $|c_{ij}|$ stands for the absolute representation value for the corresponding samples in the representation matrix $\mathbf{C}$. Cross-entropy and center loss of FC layer is:

$$\mathcal{L}_{CE} = \frac{1}{N}\sum_{j=1}^{N} ln(1 + e^{\hat{\mathbf{y}}_j^T \mathbf{q}_j}),\ \mathcal{L}_{CNT} = \frac{1}{N}\sum_{j=1}^{N}\left\|\mathbf{y}_j - \mu_{\pi(\mathbf{y}_j)}\right\|_2^2 \tag{8}$$

where $\hat{\mathbf{y}}_j^T$ stands for a softmax normalized output and $\mathbf{y}_j$ represents the logits of the FC layer for j-th sample. μ represents a centroid of a cluster π taken from the spectral clustering output for a given sample j (Zhang et al., 2019a). Additionally, in order to yield a stable and unique solution, representation matrix $\mathbf{C}$ is forced to be symmetrical (Lu et al., 2018). Symmetric loss is defined as:

$$\mathcal{L}_{SYM} = \frac{1}{2}\sum_{j=1}^{N}\|\mathbf{c}_j - \mathbf{a}_j\|_2^2 = \frac{1}{2}\|\mathbf{C} - \mathbf{A}\|^2 \tag{9}$$

where $\mathbf{A}$ represents symmetric affinity matrix. Every loss function has an assigned trade-off regularization constant λ_1 to λ_5 to regulate its importance. Thus, the total loss function is defined as follows:

$$\mathcal{L}_T = \mathcal{L}_{REC} + \lambda_1\mathcal{L}_{CIM} + \lambda_2\mathcal{L}_{CQ} + \lambda_3\mathcal{L}_{CE} + \lambda_4\mathcal{L}_{CNT} + \lambda_5\mathcal{L}_{SYM} \tag{10}$$

Furthermore, to address the overfitting issue, we present results both on the train sets and on the independent test sets. The latter will give better insights into the capability of used algorithms.

4 Experimental results

In this section, we compare the clustering performance of Robust S^2ConvSCN with state-of-the-art self-supervised S^2ConvSCN model on four well-known datasets. Because such comparison would be unfair to the Robust S^2ConvSCN model, performance is not compared to the performances of shallow and recent deep subspace clustering methods reported in the literature. Shallow models use labels to optimize hyperparameters and then evaluate the performance based on these labels. Our approach used only the in-sample labels for hyperparameter-tuning, while out-of-sample labels

were kept for evaluation. A worth-noting alternative approach of hyperparameter optimization that does not require labels was presented in (Lipor & Balzano, 2020). Regarding the label-leakage issue from pretrained parts of the network (see Figure 2), this study avoids this problem by learning from scratch within defined standards. The performance is evaluated in term of accuracy (Acc):

$$Acc(\hat{\mathbf{r}}, \mathbf{r}) = \max_{\pi \in \Pi_k} \left(\frac{1}{N} \sum_{i=1}^{N} \{\pi(\hat{\mathbf{r}}_i) = \mathbf{r}_i\} \right) \tag{11}$$

where Π_k stands for the permutation space of $[k]$. We compare the performance of Robust S^2ConvSCN with state-of-the-art self-supervised deep subspace clustering algorithm (Zhang et al., 2019a). ADAM optimizer (Kingma & Ba, 2014), an extension to stochastic gradient descent is used in the proposed learning procedure. For a full implementation details, see Appendix B.

4.1 Comparison on full datasets

As previously discussed, learning and evaluating on the same data yields an overly optimistic estimation of the model's performance. Due to the fair comparison, we implemented and compared our model to the different versions of the S^2ConvSCN model in the same manner as reported in (Zhang et al., 2019a). However, we did not use accuracy as a stopping criterion. Instead, a fixed number of epochs, learning rate decay, and early stopping based on loss decrease only were used. Thus, the obtained results differ significantly from (Zhang et al., 2019a) report. For S^2ConvSCN, optimal regularization constants were transferred from (Zhang et al., 2019a). In Table 1, a comparison of S^2ConvSCN and Robust S^2ConvSCN with different **C** matrix regularization strategies can be seen. When comparing Figure 2 and Figure 3, it can be seen that CIM prevents overtraining of the network and that loss can be used for early stopping criterion.

Table 1: Accuracy comparison of S^2ConvSCN and Robust S^2ConvSCN trained and tested on full COIL20, COIL100, and Extended YaleB datasets. BD and L2 stand for block diagonal and ℓ_2 regularization of representation matrix, respectively.

	COIL20	**COIL100**	**EYaleB**
S^{2}ConvSCN + BD	0.81111	0.30375	0.56867
S^{2}ConvSCN + L2 (Zhang et al., 2019a)	0.63333	0.55319	0.75247
RS^{2}ConvSCN + BD	0.76250	0.50528	0.41159
RS^{2}ConvSCN + L2	**0.88403**	**0.68805**	**0.81661**

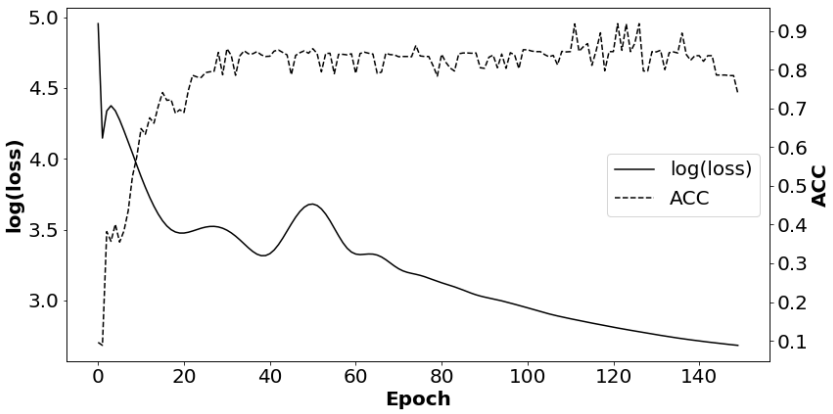

Figure 3: The figure illustrates the change of accuracy and loss over epochs for a Robust DSCNet model with CIM in the self-expressive model. In comparison with Figure 2, it can be seen that Robust DSCNet can use the change in loss as CIM prevents overfitting.

4.2 CLASSIFICATION PERFORMANCE ON UNSEEN DATA

Having several independent observations is crucial for the model's performance validation. Thus, according to Table 3, datasets were split to stratified folds. For MNIST, as there is enough data, each fold was additionally divided into a train and test set, 70% and 30%, respectively. For other datasets, the remaining folds were all used for testing as these datasets have a very limited number of data per label, i.e. additional splitting as for MNIST could not be possible. In the context of (Li et al., 2020), the obtained results in Table 2 support the claim that the network is robust to noise when early-stopping methods are applied.

Table 2: Accuracy comparison of S^2ConvSCN and Robust S^2ConvSCN trained and tested on independent folds for each dataset. BD and L2 stand for block diagonal and ℓ_2 regularization of representation matrix, respectively. The best mean accuracy for each dataset is bolded.

		MNIST	**COIL20**	**COIL100**	**EYaleB**
S^{2}ConvSCN + BD	**mean**	0.12659	0.71686	0.33762	0.49187
	stddev	0.06258	0.07814	0.04736	0.04261
S^{2}ConvSCN + L2 (Zhang et al., 2019a)	**mean**	0.12659	0.67486	0.52099	0.63735
	stddev	0.02569	0.05917	0.02295	0.07062
RS^{2}ConvSCN + BD	**mean**	**0.50417**	0.79824	0.44882	0.50017
	stddev	0.05329	0.02984	0.03845	0.02695
RS^{2}ConvSCN + L2	**mean**	0.42979	**0.82547**	**0.55461**	**0.75239**
	stddev	0.04878	0.02537	0.01454	0.03346

5 DISCUSSION AND CONCLUSION

As it could be seen in Table 2, Robust S^2ConvSCN outperforms S^2ConvSCN on all datasets regardless of regularization imposed on representation matrix $\mathbf{C}$. Moreover, in the case of the MNIST dataset, Robust S^2ConvSCN with BD regularization leads to the best results. The CIM loss, indeed, handles the corruptions in the data better than MSE, especially the possible amplification of corruptions due to their propagation through deep layers. The noise robustness can be explained by Chen et al. (2016) where is shown that learning using the correntropy-loss function generalizes better. However, training of deep learning algorithms is an especially time-consuming task. Thus, settings in Appendix Table 3 and regularization constants could possibly be improved in further testing. As discussed in earlier sections, we assume that (Zhang et al., 2019a) approach leads to an overly optimistic estimation of the model's performance. Nevertheless, Robust S^2ConvSCN outperformed S^2ConvSCN showing its superiority.

Using datasets for which the dimensionality d is unknown could be a challenging task. Without that information, it is difficult to post-process the representation matrix to eliminate the noise. Also, there is a problem with pseudo-labels refinement frequency. If it occurs too often it could lead to divergence of the whole model. If the refinement occurs too rarely, the model could get stuck in the poor minimum. Finding a more robust pseudo-label refinement strategy and elimination of the need for the prior knowledge of d is still an open question. Also, more research towards identifying which loss has higher significance during the training is needed as there are many non-equally important loss functions in Robust S^2ConvSCN. In a sense of memory constraints, mini-batch training of deep subspace clustering models could cross the barrier of memory complexity and offer more approachable learning.

To conclude, among finding a more robust model with better early stopping criterion, this ablation study aimed to set up a new, more transparent way for the evaluation of deep subspace clustering models. As baseline algorithms experience early commitment problem, i.e. they depend on weight-transfer from pretrained models, true labels are leaked during the training process. For that, we propose an evaluation of the algorithm on independent data to have a proper estimate of model performance. As can be seen from Table 2, measured performances of S^2ConvSCN significantly differ from the optimistic one presented in (Zhang et al., 2019a). The combination of gradient-based learning with CIM loss and early-stopping strategy (Chen et al., 2016; Li et al., 2020) did, indeed, improve the robustness to the unknown errors in data. Additionally, the presented model which incorporates label-free learning and the robust correntropy loss can easily be extended to multi-modal and multi-view data. We aim to address that in our further research.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Mahdi Abavisani and Vishal M Patel. Deep multimodal subspace clustering networks. *IEEE Journal of Selected Topics in Signal Processing*, 12(6):1601–1614, 2018.

Maria Brbić and Ivica Kopriva. Multi-view low-rank sparse subspace clustering. *Pattern Recognition*, 73:247–258, 2018.

Liangjun Chen, Hua Qu, Jihong Zhao, Badong Chen, and Jose C Principe. Efficient and robust deep learning with correntropy-induced loss function. *Neural Computing and Applications*, 27 (4):1019–1031, 2016.

François Chollet et al. Keras. `https://keras.io`, 2015.

Mark A Davenport, Petros T Boufounos, Michael B Wakin, and Richard G Baraniuk. Signal processing with compressive measurements. *IEEE Journal of Selected topics in Signal processing*, 4 (2):445–460, 2010.

Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

Paolo Favaro, René Vidal, and Avinash Ravichandran. A closed form solution to robust subspace estimation and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1801–1807. IEEE, 2011.

Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5736–5745, 2017.

Fulvio Gini, M Greco, and F Farina. Radar detection and preclassification based on multiple hypothesis. *IEEE transactions on aerospace and electronic systems*, 40(3):1046–1059, 2004.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, pp. 24–33, 2017.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep-subspace-clustering-networks, commit: 396c62334c1857b0e6b5bf4c4176d83fe1f806cf. `https://github.com/panji1990/Deep-subspace-clustering-networks`, 2019a.

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9865–9874, 2019b.

Mohsen Joneidi, Parvin Ahmadi, Mostafa Sadeghi, and Nazanin Rahnavard. Union of low-rank subspaces detector. *IET Signal Processing*, 10(1):55–62, 2016.

Mohsen Kheirandishfard, Fariba Zohrizadeh, and Farhad Kamangar. Multi-level representation learning for deep subspace clustering. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 2039–2048, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5): 684–698, 2005.

José Lezama, Qiang Qiu, Pablo Musé, and Guillermo Sapiro. Ole: Orthogonal low-rank embedding-a plug and play geometric loss for deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8109–8118, 2018.

Chun-Guang Li and Rene Vidal. A structured sparse plus structured low-rank framework for subspace clustering and completion. *IEEE Transactions on Signal Processing*, 64(24):6557–6570, 2016.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 4313–4324. PMLR, 2020.

John Lipor and Laura Balzano. Clustering quality metrics for subspace clustering. *Pattern Recognition*, pp. 107328, 2020.

Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2012.

Weifeng Liu, Puskal P Pokharel, and José C Príncipe. Correntropy: Properties and applications in non-gaussian signal processing. *IEEE Transactions on Signal Processing*, 55(11):5286–5298, 2007.

Muhammad Asad Lodhi and Waheed U Bajwa. Detection theory for union of subspaces. *IEEE Transactions on Signal Processing*, 66(24):6347–6362, 2018.

Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *European Conference on Computer Vision*, pp. 347–360. Springer, 2012.

Canyi Lu, Jiashi Feng, Zhouchen Lin, Tao Mei, and Shuicheng Yan. Subspace clustering by block diagonal representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2): 487–501, 2018.

Yue M Lu and Minh N Do. A theory for sampling signals from a union of subspaces. *IEEE transactions on signal processing*, 56(6):2334–2345, 2008.

S Nene, Shree K Nayar, and Hiroshi Murase. Columbia object image library (coil-100)(tech. rep.). *Columbia University*, 1996a.

Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-20). 1996b.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pp. 849–856, 2002.

Vishal M Patel and René Vidal. Kernel sparse subspace clustering. In *IEEE International Conference on Image Processing*, pp. 2849–2853. IEEE, 2014.

Vishal M Patel, Hien Van Nguyen, and René Vidal. Latent space sparse subspace clustering. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 225–232, 2013.

Xi Peng, Shijie Xiao, Jiashi Feng, Wei-Yun Yau, and Zhang Yi. Deep subspace clustering with sparsity prior. In *International Joint Conferences on Artificial Intelligence*, pp. 1925–1931, 2016a.

Xi Peng, Zhiding Yu, Zhang Yi, and Huajin Tang. Constructing the l2-graph for robust subspace learning and subspace clustering. *IEEE Transactions on Cybernetics*, 47(4):1053–1066, 2016b.

Xi Peng, Jiashi Feng, Jiwen Lu, Wei-Yun Yau, and Zhang Yi. Cascade subspace clustering. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Xi Peng, Jiashi Feng, Shijie Xiao, Wei-Yun Yau, Joey Tianyi Zhou, and Songfan Yang. Structured autoencoders for subspace clustering. *IEEE Transactions on Image Processing*, 27(10):5076–5086, 2018.

Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. Learning deep representations for graph clustering. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

René Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2011.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

Thakshila Wimalajeewa, Yonina C Eldar, and Pramod K Varshney. Subspace recovery from structured union of subspaces. *IEEE Transactions on Information Theory*, 61(4):2101–2114, 2015.

Tong Wu and Waheed U Bajwa. Revisiting robustness of the union-of-subspaces model for data-adaptive learning of nonlinear signal models. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3390–3394. IEEE, 2014.

Tong Wu and Waheed U Bajwa. Learning the nonlinear geometry of high-dimensional data: Models and algorithms. *IEEE Transactions on Signal Processing*, 63(23):6229–6244, 2015a.

Tong Wu and Waheed U Bajwa. Metric-constrained kernel union of subspaces. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5778–5782. IEEE, 2015b.

Shijie Xiao, Mingkui Tan, Dong Xu, and Zhao Yang Dong. Robust kernel low-rank representation. *IEEE Transactions on Neural Networks and Learning Systems*, 27(11):2268–2281, 2015.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, pp. 478–487, 2016.

Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4066–4075, 2019.

Han Lun Yap and Radmila Pribić. False alarms in multi-target radar detection within a sparsity framework. In *2014 International Radar Conference*, pp. 1–6. IEEE, 2014.

Junjian Zhang, Chun-Guang Li, Chong You, Xianbiao Qi, Honggang Zhang, Jun Guo, and Zhouchen Lin. Self-supervised convolutional subspace clustering network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5473–5482, 2019a.

Junjian Zhang, Chun-Guang Li, Tianming Du, Honggang Zhang, and Jun Guo. Convolutional subspace clustering network with block diagonal prior. *IEEE Access*, 8(201):5723–5732, 2020.

Tong Zhang, Pan Ji, Mehrtash Harandi, Wenbing Huang, and Hongdong Li. Neural collaborative subspace clustering. *arXiv preprint arXiv:1904.10596*, 2019b.

Lei Zhou, Bai Xiao, Xianglong Liu, Jun Zhou, Edwin R Hancock, et al. Latent distribution preserving deep subspace clustering. In *28th International Joint Conference on Artificial Intelligence*. York, 2019.

Pan Zhou, Yunqing Hou, and Jiashi Feng. Deep adversarial subspace clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1596–1604, 2018.

A APPENDIX - PRELIMINARIES

A.1 BLOCK DIAGONAL REGULARIZATION

To introduce BD regularization we state the proposition 4 from (Von Luxburg, 2007) for the graph Laplacian matrix $\mathbf{L}$:

Proposition 1. (Proposition 4 in (Von Luxburg, 2007): Number of connected components and spectra of $\mathbf{L}$). Let $\mathbf{G}$ be an undirected graph with nonnegative weights. Then the multiplicity k of the eigenvalues 0 of $\mathbf{L}$ equals the number of connected components in the graph. Thus, based on Proposition 1 we define the BD regularization of $\mathbf{C}$ as the sum of the k smallest eigenvalues of $\mathbf{L}$:

$$\|\mathbf{C}\|_{[k]} = \sum_{i=N-k+1}^{N} \lambda_i(\mathbf{L}) \tag{12}$$

where $\lambda_i(\mathbf{L})$, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{(N-k+1)} \geq \cdots \geq \lambda_N$, stands for the i-th eigenvalue of $\mathbf{L}$.

A.2 CORRENTROPY

Here we briefly introduce the correntropy and its properties that qualify it as loss function robust to data corruptions. Let $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_N] \in \mathbb{R}^{d\times N}$ and $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N] \in \mathbb{R}^{d\times N}$ be two realizations of the corresponding random variables. The empirical correntropy is estimated from data as:

$$\hat{V}(\mathbf{S}, \mathbf{T}) = \frac{1}{N}\sum_{i=1}^{N} \kappa_\sigma(\mathbf{s}_i, \mathbf{t}_i),\ \kappa_\sigma(\mathbf{s}_i, \mathbf{t}_i) = \exp\left(-\frac{\|\mathbf{s}_i - \mathbf{t}_i\|}{2\sigma^2}\right) \tag{13}$$

where $\kappa_\sigma(\mathbf{s}_i, \mathbf{t}_i)$ is the Gaussian kernel. Herein, we present from (Liu et al., 2007) two (out of ten) properties of correntropy that justify its use as robust error measure.

Property 1. (Property 3 in (Liu et al., 2007)). Correntropy involves all the even moments of the random variable $\boldsymbol{\varepsilon} = \mathbf{T} - \mathbf{S}$:

$$V_\sigma(\mathbf{S}, \mathbf{T}) = \frac{1}{\sqrt{2\pi}\sigma}\sum_{n=0}^{\infty} \frac{(-1)^n}{2^n n!}\mathbb{E}\left[\frac{(\mathbf{S}-\mathbf{T})^{2n}}{\sigma^{2n}}\right] \tag{14}$$

where $\mathbb{E}$ denotes mathematical expectation. As σ increases, the high-order moments decay faster; so, the second-order moment tends to dominate, and the correntropy approaches correlation.

While the MSE involves only second-order moments and is, thus, optimal measure for error distributed normally, correntropy is the optimal measure for error with the arbitrary (non-Gaussian) distribution. Furthermore, as can be seen from Equation (13), correntropy is data-driven.

Property 2 (Property 8 in (Liu et al., 2007)). The function:

$$CIM(\mathbf{S}, \mathbf{T}) = \left(\kappa(\mathbf{0}, \mathbf{0}) - \hat{V}(\mathbf{S}, \mathbf{T})\right)^{\frac{1}{2}} \tag{15}$$

defines a CIM in sample space. Since its limes is final, the CIM function is robust to large random errors.

B APPENDIX - EXPERIMENTAL SETUP

Robust S^2ConvSCN model is implemented in Keras (Chollet et al., 2015) and Tensorflow (Abadi et al., 2015). First, the random seed is fixed for reproducibility. To have a reliable performance estimate on the test set and multiple independent train and test folds for each dataset, a stratified k-fold splitting was performed. Depending on the dataset constraints, the train-test split was performed on every fold or one fold served as a train set while the rest $k-1$ folds served as a test set. Hence, independent k observations of algorithms' performances for each dataset have been produced.

As discussed in Section 3, instead of monitoring accuracy every epoch, and consequently supervising the learning using ground truth labels, we use only the loss for reducing learning rate and early stopping. The training is stopped either after reaching the early stopping criterion or after reaching the maximum number of epochs. As in (Zhang et al., 2019a), pretraining was also applied. Firstly, the autoencoder is pretrained to replicate the input. Secondly, the self-expression layer together with the pretrained autoencoder (DSCNet (Ji et al., 2017)) was trained to reach a limited number of epochs or the early stopping criterion. After that, the FC layer and self-supervision modules were added to the pretrained DSCNet. This procedure forms a starting point for both S^2ConvSCN and Robust S^2ConvSCN.

It is important to note that, once trained, the **C** matrix from the self-expression layer is post-processed in two steps, as done in DSCNet (Ji et al., 2019a). First, it is thresholded by keeping only δ largest, in terms of magnitude, elements in a row. The accepted number of largest elements depends on their sum - S_δ. If the sum S_δ exceeds $\lambda \sum_{i=0}^{N} c_{ji},\ i \in \{0, \ldots, N\}$, where λ stands for an empirically set thresholding constant and N represents the number of elements in a row, other elements (not included in the sum S_δ) are set to zero. Second, knowing the dimensionality of the dataset d, after SVD decomposition of thresholded representation matrix **C** only d largest eigenvalues with related eigenvectors were kept meaning the remaining eigenvectors span the noise subspace. The reconstructed representation matrix serves as an input to the spectral clustering algorithm which produces pseudo-labels.

During the warming-up phase, pseudo-labels were not refined because the newly introduced layers and modules could affect the convergence process (see Section 3 for the discussion). For the sake of fair comparison, different versions of S^2ConvSCN were trained using the same settings as Robust S^2ConvSCN. Table 3 shows settings for MINST (LeCun et al., 2010), COIL-20 (Nene et al., 1996b), COIL-100 (Nene et al., 1996a), and Extended Yale B (Lee et al., 2005).

Table 3: Settings of Robust S^2ConvSCN per each dataset. T-max represents maximum number epochs and T0 stands for the number of epochs after which pseudo-labels are refined. Warm-up represents the number of epochs without the refinement of pseudo-labels, LR stands for starting learning rate, and Min LR represents the minimum learning rate.

	MNIST	**COIL20**	**COIL100**	**EYaleB**
data splits	15	5	5	5
T-max	9000	4000	4000	9000
T0	30	50	40	30
Warm-up	50	100	80	50
LR	1e-3	1e-4	1e-4	1e-4
Min LR	1e-6	1e-6	1e-7	1e-6

C APPENDIX - COMPLEXITY AND CONVERGENCE

Dataset size and choice of representation matrix regularization have a direct impact on time and memory complexity. As the model must learn in batch mode, the number of samples in the dataset N determines the number of parameters in the self-expression layer - N^2. Thus, dataset size impacts memory and time complexity. Also, regularization imposed on the representation matrix affects the computational time of the algorithm. As presented in Section A, BD regularization is defined as the sum of k smallest eigenvalues of Laplacian matrix **L**. Thus, if the BD regularization is chosen,

the model will have higher time complexity due to the usage of the singular value decomposition algorithm. Although it is not directly associated with the time complexity per epoch, the spectral clustering module refines pseudo-labels every $T0$ epochs which adds to overall training time.

To properly emphasize individual loss functions, all regularization constants in the overall loss function must be tuned (discussed in Section 3). As a good starting point, we used regularization constants from (Zhang et al., 2019a) and refined only those associated with the error measure term in the self-expression layer and representation matrix $\mathbf{C}$. Depending on the fold size and regularization used, training of one fold on one GPU takes 2-6 hours. Testing on one fold is within several seconds.