Supplementary Material – On the Effects of Self-supervision and Contrastive Alignment in Deep Multi-view Clustering

Daniel J. Trosten[*], Sigurd Løkse[*], Robert Jenssen[*†‡§], Michael C. Kampffmeyer[*†]
Department of Physics and Technology, UiT The Arctic University of Norway
firstname[.middle initial].lastname@uit.no

1. Introduction

Here, we provide the proofs for Propositions 2 and 3; additional details on the proposed new instances of DeepMVC; the datasets used for evaluation; the hyperparameters used by baselines and new instances; and the computation of metrics and uncertainties used in our evaluation protocol. We also include the full list of recent methods and their DeepMVC components, the complete table of results from the experimental evaluation. In addition, we include additional experiments and analyses of reproducibility, hyperparameters, and the Fusion and CM components. Finally, we reflect on possible negative societal impacts of our work.

Our implementation of the DeepMVC framework, as well as the datasets and the evaluation protocol used in our experiments, is available at https://github.com/DanielTrosten/DeepMVC. See README.md in the repository for more details about the implementation, and how to reproduce our results.

2. Previous methods as instances of DeepMVC

The full list of recent methods and their DeepMVC components is given in Table 1. We observe that all but one model includes at least one form of SSL, but the type of SSL, and also fusion and CM, vary significantly for the different models. This illustrates the importance of the SSL components in deep MVC, as well as the need for a unified framework with a consistent evaluation protocol, in order to properly compare and evaluate methods.

3. Contrastive alignment in deep MVC

Proof of propositions

Proposition 2. *Suppose $k_v, v \in \mathbb{N}$ are random variables taking values in $\{1, \dots, k\}$. Then, for any $V \geq 1$,*

$$\mathbb{P}\{\min_{v=1,\dots,V+1}\{k_v\} \leq \min_{v=1,\dots,V}\{k_v\} \;\Big|\; k_1, \dots, k_V\} = 1 \tag{1}$$

Proof. Let $M_V = \min_{v=1,\dots,V}\{k_v\}$, then we need to prove that

$$\mathbb{P}(M_{V+1} \leq M_V \mid k_1, \dots, k_V) = 1. \tag{2}$$

Due to the properties of the minimum operator, we have

$$\begin{cases} M_{V+1} = M_V, & \text{if } k_{V+1} \geq M_V \\ M_{V+1} < M_V, & \text{otherwise} \end{cases}. \tag{3}$$

Hence, $M_{V+1} \leq M_V$ regardless of the value of k_{V+1}, which gives

$$\mathbb{P}(M_{V+1} \leq M_V \mid k_1, \dots, k_V) = 1. \tag{4}$$

□

Proposition 3. *Suppose $k_v, v \in \mathbb{N}$ are iid. random variables taking values in $\{1, \dots, k\}$. Then, for any $V \geq 1$,*

$$\mathbb{E}(\min_{v=1,\dots,V+1}\{k_v\}) \leq \mathbb{E}(\min_{v=1,\dots,V}\{k_v\}) \tag{5}$$

Proof. Let $M_V = \min_{v=1,\dots,V}\{k_v\}$, then

$$F_{M_V}(x) \coloneqq \mathbb{P}(M_V \leq x) = 1 - \mathbb{P}(M_V > x) \tag{6}$$
$$= 1 - \mathbb{P}(k_1 > x \cap \dots \cap k_V > x) \tag{7}$$
$$= 1 - (1 - F_{k_v}(x))^V \tag{8}$$

where $F_{k_v}(x) = \mathbb{P}(k_v \leq x)$.

Since M_V is a non-negative random variable, we have

$$\mathbb{E}(M_V) = \sum_{x=0}^{\infty}(1 - F_{M_V}(x)) = \sum_{x=0}^{\infty}(1 - F_{k_v}(x))^V. \tag{9}$$

[*]UiT Machine Learning group (machine-learning.uit.no) and Visual Intelligence Centre (visual-intelligence.no).
[†]Norwegian Computing Center (nr.no).
[‡]Department of Computer Science, University of Copenhagen.
[§]Pioneer Centre for AI (aicentre.dk).

Model	Pub.	Enc.	SV-SSL	MV-SSL	Fusion	CM
DCCAE [17]	ICML'15	MLP	Reconstruction	CCA	1^{st} view	SC
DMSC [1]	J. STSP'18	CNN	Reconstruction	–	Affinity fusion	SR, SC
DMVSSC [13]	ICNCC'18	CNN	Reconstruction	–	–	Sparse SR, SC
MvSN [4]	T. CSS'19	MLP	Sp. Emb.	–	Weighted sum	k-means
MvSCN [5]	IJCAI'19	MLP	Sp. Emb.	MSE Al.	Concat.	k-means
MvDSCN [26]	arXiv'19	CNN	–	Reconstruction	Shared network	SR, SC
DAMC [9]	IJCAI'19	MLP	–	Reconstruction	Average	DEC
S2DMVSC [12]	ACML'19	MLP	Reconstruction	–	MLP	SR, SC
DCMR [24]	PAKDD'20	MLP	Variational Reconstruction	Variational Reconstruction	MLP	k-means
DMMC [23]	ICME'20	MLP	Reconstruction	–	MLP	Fusion output
DCUMC [27]	ICIKM'20	MLP	Reconstruction	Commonness uniqueness	MLP	k-means
SGLR-MVC [22]	AAAI'20	MLP	–	Variational Reconstruction	Weighted sum	GMM
EAMC [25]	CVPR'20	MLP	–	Distribution Al., Kernel Al.	Attention	DDC
MVC-MAE [2]	DSE'21	MLP	Reconstruction, Ngh. preserv.	Contrastive Al.	–	DEC
SDC-MVC [19]	IJCNN'21	MLP	–	CCA	Concat.	DEC
DEMVC [20]	Inf. Sci.'21	CNN	Reconstruction	–	–	DEC
SiMVC [14]	CVPR'21	MLP/ CNN	–	–	Weighted sum	DDC
CoMVC [14]	CVPR'21	MLP/ CNN	–	Contrastive Al.	Weighted sum	DDC
Multi-VAE [21]	ICCV'21	CNN	–	Variational Reconstruction	Concat.	Gumbel, k-means
DMIM [10]	IJCAI'21	MLP	Min. superflous information	Max. shared information	?	Encoder output
AMvC [15]	TNNLS'22	MLP	–	Reconstruction	Weighted sum	DEC
SIB-MSC [16]	arXiv'22	CNN	–	Reconstruction, Inf. Bottleneck	Affinity fusion	SR, SC

Abbreviations: "–" = Not included, "?" = Not specified, Al. = Alignment, Concat. = Concatenate, CCA = Canonical correlation analysis, DDC = Deep divergence-based clustering, DEC = Deep embedded clustering, Inf. Bottleneck = Information bottleneck, Ngh. preserv. = Neighborhood preservation, SC = Spectral clustering, Sp. Emb. = Spectral Embedding, SR = Self-representation, Sparse SR = Sparse self-representation,

Table 1. Full overview of methods from previous work and their DeepMVC components.

Hence

$$\begin{aligned}
&\mathbb{E}(M_V) - \mathbb{E}(M_{V+1}) \\
&= \sum_{x=0}^{\infty}(1 - F_{k_v}(x))^V - \sum_{x=0}^{\infty}(1 - F_{k_v}(x))^{V+1} && (10)\\
&= \sum_{x=0}^{\infty}(1 - F_{k_v}(x))^V(1 - (1 - F_{k_v}(x))) && (11)\\
&= \sum_{x=0}^{\infty}\underbrace{(1 - F_{k_v}(x))^V}_{\geq 0}\underbrace{F_{k_v}(x)}_{\geq 0} \geq 0 && (12)
\end{aligned}$$

which is a sum of non-negative terms, since $F_{k_v}(x) \in [0, 1]$ is a probability. This gives

$$\mathbb{E}(M_{V+1}) \leq \mathbb{E}(M_V) \quad (13)$$

□

4. New instances of DeepMVC

In this section we provide additional details on loss functions, particularly the weighted sum fusion, and the DDC [7] clustering module. The loss functions used to train the new instances are on the form

$$\mathcal{L}^{\text{Total}} = w^{\text{SV}}\mathcal{L}^{\text{SV}} + w^{\text{MV}}\mathcal{L}^{\text{MV}} + w^{\text{CM}}\mathcal{L}^{\text{CM}} \quad (14)$$

where $\mathcal{L}^{\text{SV}}$, $\mathcal{L}^{\text{MV}}$, and $\mathcal{L}^{\text{CM}}$ denote the losses from the SV-SSL, MV-SSL, and CM components, respectively. Note that the losses $\mathcal{L}^{\text{SV}}$ and $\mathcal{L}^{\text{MV}}$ correspond to the losses in Section 5 of the main paper. $(w^{\text{SV}}, w^{\text{MV}}, w^{\text{CM}})$ are optional weights for the respective losses, which are all set to 1 unless specified otherwise.

Connection between InfoDDC and contrastive self-supervised learning For two views $u \neq v \in 1, \ldots, V$, contrastive SSL can be regarded as variational maximization of the mutual information

$$I(\boldsymbol{z}^{(v)}, \boldsymbol{z}^{(u)}) \tag{15}$$

where $\boldsymbol{z}^{(v)}$ and $\boldsymbol{z}^{(u)}$ have *multi-variate, continuous* distributions in $\mathbb{R}^d$.

In InfoDDC, we instead maximize mutual information between pairs of *uni-variate, discrete* random variables

$$I(c^{(v)}, c^{(u)}) \tag{16}$$

where we assume that the distributions of $c^{(v)}$ and $c^{(u)}$ are given by the view-specific representations

$$\mathbb{P}(c^{(w)} = i) = z^{(w)}_{[i]}, \quad i = 1, \ldots, d, \quad w \in \{u, v\} \tag{17}$$

where $z^{(w)}_{[i]}$ denotes component i of the view-specific representation $\boldsymbol{z}^{(w)} = f^{(w)}(\boldsymbol{x}^{(w)})$. Hence, although InfoDDC might appear similar to CA-based methods, the maximization of mutual information is done for different pairs of random variables.

Weighted sum fusion. As [14], we implement the weighted sum fusion as

$$\boldsymbol{z}_i = \sum_{v=1}^{V} w^{(v)} \boldsymbol{z}_i^{(v)}, \tag{18}$$

where the weights $w^{(1)}, \ldots, w^{(V)}$ are non-negative and sum to 1. These constraints are implemented by keeping a vector of trainable, un-normalized weights, from which $w^{(1)}, \ldots, w^{(V)}$ can be computed by applying the softmax function.

DDC clustering module. The DDC [7] clustering module consists of two fully-connected layers. The first layer calculates the hidden representation $\boldsymbol{h}_i \in \mathbb{R}^{D_{DDC}}$ from the fused representation $\boldsymbol{z}_i$. The dimensionality of the hidden representation, D_{DDC} is a hyperparameter set to 100 for all models. The second layer computes the cluster membership vector $\boldsymbol{\alpha}_i \in \mathbb{R}^k$ from the hidden representation.

DDC's loss function consists of three terms

$$\mathcal{L}^{\text{CM}}_{\text{DDC}} = \mathcal{L}_{\text{DDC, 1}} + \mathcal{L}_{\text{DDC, 2}} + \mathcal{L}_{\text{DDC, 3}}. \tag{19}$$

The three terms encourage (i) separable and compact clusters in the hidden space; (ii) orthogonal cluster membership vectors; and (iii) cluster membership vectors close to simplex corners, respectively.

The first term maximizes the pairwise Cauchy-Schwarz divergence [6] between clusters (represented as probability densities) in the space of hidden representations

$$\mathcal{L}_{\text{DDC, 1}} = \tag{20}$$

$$\binom{k}{2}^{-1} \sum_{a=1}^{k-1} \sum_{b=a}^{k} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ia} \kappa_{ij} \alpha_{jb}}{\sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ia} \kappa_{ij} \alpha_{ja} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ib} \kappa_{ij} \alpha_{jb}}} \tag{21}$$

where $\kappa_{ij} = \exp\left(-\frac{||\boldsymbol{h}_i - \boldsymbol{h}_j||^2}{2\sigma^2}\right)$ and σ is a hyperparameter. Following [7], we set σ to 15% of the median pairwise difference between the hidden representations.

The second term minimizes the pairwise inner product between cluster membership vectors

$$\mathcal{L}_{\text{DDC, 2}} = \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j^{\top}. \tag{22}$$

The third term encourages cluster membership vectors to be close to the corners of the probability simplex in $\mathbb{R}^k$

$$\mathcal{L}_{\text{DDC, 3}} = \tag{23}$$

$$\binom{k}{2}^{-1} \sum_{a=1}^{k-1} \sum_{b=a}^{k} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} m_{ia} \kappa_{ij} m_{jb}}{\sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} m_{ia} \kappa_{ij} m_{ja} \sum_{i=1}^{n} \sum_{j=1}^{n} m_{ib} \kappa_{ij} m_{jb}}} \tag{24}$$

where $m_{ia} = \exp(-||\boldsymbol{\alpha}_i - \boldsymbol{e}_a||^2)$, and $\boldsymbol{e}_a$ is the a-th simplex corner.

5. Experiments

5.1. Datasets

Dataset details are listed in Table 2. The code repository includes pre-processed Caltech7 and Caltech20 datasets. The other datasets can be generated by following the instructions in `README.md` (these could not be included in the archive due to limitations on space).

Caltech details. We use the same features and subsets of the Caltech101 [3] dataset as [5].

- **Features:** Gabor, Wavelet Moments, CENsus TRansform hISTogram (CENTRIST), Histogram of Oriented Gradients (HOG), GIST, and Local Binary Patterns (LBP).
- **Caltech7 classes:** Face, Motorbikes, Dolla-Bill, Garfield, Snoopy, Stop-Sign, Windsor-Chair.

Dataset	n	v	k	n_{small}	n_{big}	Dim.	Licence
NoisyMNIST [8]	70000	2	10	6313	7877	$(28 \times 28)^2$	CC BY-SA 3.0
NoisyFashion [18]	70000	2	10	7000	7000	$(28 \times 28)^2$	MIT
EdgeMNIST [8]	70000	2	10	6313	7877	$(28 \times 28)^2$	CC BY-SA 3.0
EdgeFashion [18]	70000	2	10	7000	7000	$(28 \times 28)^2$	MIT
COIL-20 [11]	480	3	20	24	24	$(64 \times 64)^3$	None
Caltech7 [3]	1474	6	7	34	798	$48, 40, 254, 1984, 512, 928$	CC BY 4.0
Caltech20 [3]	2386	6	20	33	798	$48, 40, 254, 1984, 512, 928$	CC BY 4.0
PatchedMNIST [8]	21770	12	3	6903	7877	$(28 \times 28)^{12}$	CC BY-SA 3.0

Table 2. Dataset details. n = number of instances, v = number of views, k = number of classes/clusters, n_{small} = number of instances in smallest class, n_{big} = number of instances in largest class, Dim. = view dimensions.

- **Caltech20 classes:** Face, Leopards, Motorbikes, Binocular, Brain, Camera,Car-Side, Dolla-Bill, Ferry, Garfield, Hedgehog, Pagoda, Rhino, Snoopy, Stapler, Stop-Sign, Water-Lilly, WindsorChair, Wrench, Yin-yang.

5.2. Hyperparameters

Network architectures. The encoder and decoder architectures are listed in Table 3. MLP encoders/decoders are used for Caltech7 and Caltech20 as these contain vector data. The other datasets contain images, so CNN encoders and decoders are used for them.

Other hyperparameters. Table 4 lists other hyperparameters used for the baselines and new instances.

5.3. Computational resources

We run our experiments on a Kubernetes cluster, where jobs are allocated to nodes with Intel(R) Xeon(R) E5-2623 v4 or Intel(R) Xeon(R) Silver 4210 CPUs (2 cores allocated per job); and Nvidia GeForce GTX 1080 Ti or Nvidia GeForce RTX 2080 Ti GPUs. Each job has 16 GB RAM available.

With this setup, 5 training runs on NoisyMNIST, NoisyFashion, EdgeMNIST, and EdgeFashion take approximately 24 hours. Training times for the other datasets are approximately between 1 and 3 hours.

The Dockerfile used to build our docker image can be found in the code repository.

5.4. Evaluation protocol

Metrics. We measure performance using the accuracy

$$\text{ACC} = \max_{m \in \mathcal{M}} \frac{\sum_{i=1}^{n} \delta(m(\hat{y}_i) - y_i)}{n} \tag{25}$$

where $\delta(\cdot)$ is the Kronecker-delta, $\hat{y}_i$ is the predicted cluster of instance i, and y_i is the ground truth label of instance i. The maximum runs over $\mathcal{M}$, which is the set of all bijective mappings from $\{1, \ldots, k\}$ to itself.

We also compute the normalized mutual information

$$\text{NMI} = \frac{MI(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\frac{1}{2}(H(\hat{\boldsymbol{y}}) + H(\boldsymbol{y}))} \tag{26}$$

where $\hat{\boldsymbol{y}} = [\hat{y}_1, \ldots, \hat{y}_n]$, $\boldsymbol{y} = [y_1, \ldots, y_n]$, $MI(\cdot, \cdot)$ and $H(\cdot)$ denotes the mutual information and entropy, respectively.

Uncertainty estimation. The uncertainty of our performance statistic can be estimated using bootstrapping. Suppose the R training runs result in the R tuples

$$(L_1, M_1), \ldots, (L_R, M_R) \tag{27}$$

where L_i is the final loss of run i, and M_i is resulting performance metric for run i. We then sample B bootstrap samples uniformly from the original results

$$(L_j^b, M_j^b) \sim \text{Uniform}\{(L_1, M_1), \ldots, (L_R, M_R)\}, \quad j = 1, \ldots, R, \quad b = 1, \ldots B. \tag{28}$$

The performance statistic for bootstrap sample b is then given by

$$M_\star^b = M_{j_\star^b}^b, \quad j_\star^b = \arg\min_{j=1,\ldots,R} \{L_j^b\}. \tag{29}$$

We then estimate the uncertainty of the performance statistic by computing the standard deviation of the bootstrap statistics $M_\star^1, \ldots M_\star^B$

$$\hat{\sigma}_{M_\star} = \sqrt{\frac{\sum_{b=1}^{B}(M_\star^b - \bar{M}_\star)^2}{B-1}}, \text{ where } \quad \bar{M}_\star = \frac{\sum_{b=1}^{B} M_b^\star}{B}. \tag{30}$$

5.5. Results

Evaluation results. The complete evaluation results are given in Table 5.

Ablation study – Fusion and Clustering module. Table 6 shows the results of ablation studies with the fusion and clustering module (CM) components. Since these components can not be completely removed, we instead replace more

CNN encoder	CNN decoder	MLP encoder	MLP decoder
`Conv`$(64 \times 3 \times 3)$	`UpSample`(2×2)	`Dense`(1024)	`Dense`(256)
`ReLU`	`TransposeConv`$(64 \times 3 \times 3)$	`BatchNorm`	`BatchNorm`
`Conv`$(64 \times 3 \times 3)$	`ReLU`	`ReLU`	`ReLU`
`BatchNorm`	`TransposeConv`$(64 \times 3 \times 3)$	`Dense`(1024)	`Dense`(1024)
`ReLU`	`BatchNorm`	`BatchNorm`	`BatchNorm`
`MaxPool`(2×2)	`ReLU`	`ReLU`	`ReLU`
`Conv`$(64 \times 3 \times 3)$	`UpSample`(2×2)	`Dense`(1024)	`Dense`(1024)
`ReLU`	`TransposeConv`$(64 \times 3 \times 3)$	`BatchNorm`	`BatchNorm`
`Conv`$(64 \times 3 \times 3)$	`ReLU`	`ReLU`	`ReLU`
`BatchNorm`	`TransposeConv`$(1 \times 3 \times 3)$	`Dense`(1024)	`Dense`(1024)
`ReLU`	`Sigmoid`	`BatchNorm`	`BatchNorm`
`MaxPool`(2×2)		`ReLU`	`ReLU`
		`Dense`(256)	`Dense(input dim)`
			`Sigmoid`

Table 3. Network architectures.

Model	Batch size	Learning rate	w^{SV}	w^{MV}	w^{CM}	Pre-train	Gradient clip
DMSC	100	10^{-3}	1.0	–	–	✓	10
MvSCN	512	10^{-4}	0.999	0.001	–	✘	10
EAMC	100	†	–	1.0	1.0	✘	10
SiMVC	100	10^{-3}	–	–	1.0	✘	10
CoMVC	100	10^{-3}	–	0.1	1.0	✘	10
Multi-VAE	64	$5 \cdot 10^{-4}$	–	1.0	–	✓	10
AE–DDC	100	10^{-3}	1.0	–	1.0	✓	10
AECoDDC	100	10^{-3}	1.0	0.1	1.0	✓	10
AE–KM	100	10^{-3}	1.0	–	–	✘	10
AECoKM	100	10^{-3}	1.0	0.1	–	✘	10
InfoDDC	256	10^{-3}	–	0.1	1.0	✘	10
MV-IIC	256	10^{-3}	–	0.01	1.0	✘	10

Table 4. Hyperparameters used to train the models. † = EAMC [25] has different learning rates for the different components, namely 10^{-5} for the encoders and clustering module, and 10^{-4} for the attention module and discriminator.

complicated components, with the simplest possible component. Thus, we replace weighted sum with concatenate for the fusion component, and DDC with k-means for the CM component.

For the fusion component, we see that the weighted sum tends to improve over the concatenation. For the CM, we observe that the performance is better with DDC than with k-means on NoisyMNIST, but the improvement more varied on Caltech7. This is consistent with what we observed in the evaluation results in the main paper.

Reproducibility of original results. Table 7 compares the results of our re-implementation of the baselines, to the results reported by the original authors. The comparison shows large differences in performance for several methods, and the differences are particularly large for MvSCN and Multi-VAE. For MvSCN, we do not use the same autoencoder preprocessing of the data. We also had difficulties getting the Cholesky decomposition to converge during training. For MultiVAE, we note that NoisyMNIST and NoisyFashion are generated without noise in the original paper, possibly resulting in datasets that are simpler to cluster. We were however not able to determine the reason for the difference in performance on COIL-20.

Additionally, all methods use different network architectures and evaluation protocols in the original publications, making it difficult to accurately compare performance between methods and their implementations. This illustrates the difficulty of reproducing and comparing results in deep MVC, highlighting the need for a unified framework with a consistent evaluation protocol and an open-source implementation.

Sensitivity to hyperparameters Table 8 shows the results of hyperparameter sweeps for the following hyperparameters:

- Weight of reconstruction loss (w^{SV}).
- Weight of contrastive loss (w^{MV}).

	NoisyMNIST		NoisyFashion		EdgeMNIST		EdgeFashion	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
DMSC	0.66 (0.02)	0.67 (0.01)	0.49 (0.05)	0.48 (0.03)	0.51 (0.02)	0.47 (0.02)	0.52 (0.01)	0.47 (0.00)
MvSCN	0.15 (0.00)	0.02 (0.00)	0.14 (0.00)	0.01 (0.00)	0.14 (0.00)	0.01 (0.01)	0.12 (0.00)	0.03 (0.00)
EAMC	0.83 (0.04)	0.90 (0.02)	0.61 (0.02)	0.71 (0.02)	0.76 (0.05)	0.79 (0.03)	0.51 (0.03)	0.47 (0.01)
SiMVC	**1.00** (0.02)	**1.00** (0.02)	0.52 (0.02)	0.51 (0.02)	0.89 (0.06)	0.90 (0.04)	0.61 (0.01)	0.56 (0.02)
CoMVC	**1.00** (0.00)	**1.00** (0.00)	0.67 (0.03)	0.68 (0.03)	**0.97** (0.08)	**0.94** (0.07)	0.56 (0.03)	0.52 (0.01)
Multi-VAE	0.98 (0.05)	0.96 (0.02)	0.62 (0.02)	0.60 (0.01)	0.85 (0.01)	0.76 (0.01)	0.58 (0.01)	**0.64** (0.00)
AE–KM	0.74 (0.03)	0.71 (0.00)	0.58 (0.02)	0.59 (0.01)	0.60 (0.00)	0.57 (0.00)	0.54 (0.00)	0.58 (0.00)
AE–DDC	**1.00** (0.04)	**1.00** (0.03)	0.69 (0.06)	0.65 (0.05)	0.88 (0.11)	0.88 (0.09)	0.60 (0.01)	0.58 (0.01)
AECoKM	**1.00** (0.00)	0.99 (0.00)	0.63 (0.07)	0.73 (0.03)	0.38 (0.03)	0.31 (0.02)	0.39 (0.04)	0.34 (0.02)
AECoDDC	**1.00** (0.00)	0.99 (0.00)	**0.80** (0.02)	**0.77** (0.01)	0.89 (0.10)	0.90 (0.09)	**0.67** (0.09)	0.62 (0.06)
InfoDDC	0.90 (0.05)	0.92 (0.04)	0.54 (0.03)	0.52 (0.04)	0.62 (0.04)	0.52 (0.06)	0.43 (0.01)	0.43 (0.03)
MV-IIC	0.52 (0.04)	0.79 (0.02)	0.52 (0.07)	0.74 (0.02)	0.31 (0.04)	0.21 (0.05)	0.52 (0.04)	0.59 (0.04)

	COIL-20		Caltech7		Caltech20		PatchedMNIST	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
DMSC	$-^{\dagger}$ (–)	$-^{\dagger}$ (–)	0.50 (0.03)	0.50 (0.02)	0.35 (0.01)	0.55 (0.00)	$-^{\dagger}$ (–)	$-^{\dagger}$ (–)
MvSCN	0.21 (0.00)	0.23 (0.01)	0.29 (0.02)	0.02 (0.00)	0.13 (0.01)	0.09 (0.01)	$-^{\dagger}$ (–)	$-^{\dagger}$ (–)
EAMC	0.39 (0.15)	0.52 (0.22)	0.44 (0.02)	0.23 (0.03)	0.22 (0.04)	0.23 (0.02)	$-^{\ddagger}$ (–)	$-^{\ddagger}$ (–)
SiMVC	**0.90** (0.04)	**0.96** (0.02)	0.41 (0.02)	0.51 (0.09)	0.34 (0.02)	0.52 (0.01)	0.84 (0.04)	0.64 (0.11)
CoMVC	0.87 (0.03)	**0.96** (0.02)	0.38 (0.01)	0.55 (0.02)	0.34 (0.01)	0.59 (0.02)	0.73 (0.12)	0.57 (0.19)
Multi-VAE	0.74 (0.02)	0.84 (0.01)	0.47 (0.02)	0.47 (0.01)	0.40 (0.01)	0.57 (0.01)	0.94 (0.00)	0.77 (0.00)
AE–KM	0.88 (0.04)	0.92 (0.01)	0.44 (0.03)	0.52 (0.01)	0.45 (0.02)	0.57 (0.01)	0.87 (0.00)	0.68 (0.01)
AE–DDC	0.80 (0.04)	0.93 (0.02)	0.40 (0.01)	0.54 (0.07)	0.34 (0.01)	0.44 (0.03)	0.77 (0.10)	0.59 (0.17)
AECoKM	0.84 (0.04)	0.94 (0.02)	0.20 (0.01)	0.05 (0.00)	0.22 (0.02)	0.27 (0.02)	0.96 (0.00)	0.85 (0.00)
AECoDDC	0.87 (0.01)	**0.96** (0.00)	0.36 (0.01)	0.43 (0.03)	0.31 (0.02)	0.51 (0.02)	**0.99** (0.00)	**0.97** (0.00)
InfoDDC	0.25 (0.04)	0.54 (0.03)	0.51 (0.01)	0.60 (0.04)	**0.58** (0.07)	**0.63** (0.03)	**0.99** (0.00)	0.96 (0.00)
MV-IIC	0.83 (0.05)	0.94 (0.02)	**0.53** (0.00)	**0.63** (0.04)	0.49 (0.01)	0.61 (0.01)	0.97 (0.00)	0.90 (0.01)

Table 5. Clustering results. Standard deviations (obtained by bootstrapping) are shown in parentheses. † = training ran out of memory, ‡ = training resulted in NaN loss.

- Temperature in contrastive loss (τ).

- Weight of entropy regularization (λ).

We emphasize that these results were *not* used to tune hyperparameters for the new instances. Rather, they are included to investigate how robust these methods are towards changes in the hyperparameter configuration. The results show that the new instances are mostly insensitive to changes in their hyperparameters. We however observe two cases where the hyperparameter configurations can have significant impact on the model performance. First, AECoDDC shows a drop in performance when the weight of the contrastive loss is set to high on Caltech7 (Table 8b). This is consistent with our observations regarding contrastive alignment on datasets with many views. Second, InfoDDC and MV-IIC performs worse when the entropy regularization weight is set too low, indicating that sufficient regularization is required for these models to perform well.

6. Potential negative societal impacts

As is the case with most methodological research, our work can be applied to downstream applications with negative societal impact – for instance by reflecting biases in the dataset the model was trained on. We note that in unsupervised learning, it is particularly important to check what a model has learned, due to the lack of label supervision. This is crucial if the models are used to make high-stakes decisions.

References

[1] Mahdi Abavisani and Vishal M. Patel. Deep Multimodal Subspace Clustering Networks. *IEEE Journal of Selected Topics in Signal Processing*, 12(6):1601–1614, 2018. 2

[2] Guowang Du, Lihua Zhou, Yudi Yang, Kevin Lü, and Lizhen Wang. Deep Multiple Auto-Encoder-Based Multi-view Clustering. *Data Science and Engineering*, 6(3):323–338, 2021. 2

[3] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories.

(a) Fusion

Model	NoisyMNIST		Caltech7	
	Concat.	Weighted	Concat.	Weighted
SiMVC	1.00	1.00 (0.00)	0.36	0.41 (+0.04)
CoMVC	1.00	1.00 (0.00)	0.42	0.38 (−0.04)
AE–DDC	1.00	1.00 (0.00)	0.36	0.40 (+0.04)
AECoDDC	1.00	1.00 (0.00)	0.39	0.36 (−0.03)
InfoDDC	0.93	0.90 (−0.03)	0.36	0.51 (+0.15)

(b) CM

Model	NoisyMNIST		Caltech7	
	k-means	DDC	k-means	DDC
SiMVC	0.67	1.00 (+0.33)	0.39	0.41 (+0.01)
CoMVC	0.56	1.00 (+0.44)	0.22	0.38 (+0.16)
AE–DDC	0.74	1.00 (+0.26)	0.44	0.40 (−0.04)
AECoDDC	1.00	1.00 (0.00)	0.20	0.36 (+0.16)
InfoDDC	0.14	0.90 (+0.76)	0.59	0.51 (−0.08)

Table 6. Accuracies from ablation studies with the Fusion and CM components.

Model	Dataset	Orig.	Ours
MvSCN	N-MNIST	0.99	0.15 (−0.84)
	Caltech20	0.59	0.13 (−0.46)
EAMC	E-MNIST	0.67	0.76 (+0.09)
SiMVC	E-MNIST	0.86	0.89 (+0.03)
	E-Fashion	0.57	0.61 (+0.04)
	COIL-20	0.78	0.90 (+0.12)
CoMVC	E-MNIST	0.96	0.97 (+0.01)
	E-Fashion	0.60	0.56 (−0.04)
	COIL-20	0.89	0.87 (−0.02)
Multi-VAE	N-MNIST†	1.00	0.98 (−0.02)
	N-Fashion†	0.91	0.62 (−0.29)
	COIL-20	0.98	0.74 (−0.24)

Table 7. Accuracies from our experiment vs. accuracies reported by the original authors. † = method is originally evaluated on a slightly different dataset.

Computer Vision and Image Understanding, 106(1):59–70, 2007. 3, 4

[4] Shuning Huang, Kaoru Ota, Mianxiong Dong, and Fanzhang Li. MultiSpectralNet: Spectral Clustering Using Deep Neural Network for Multi-View Data. *IEEE Transactions on Computational Social Systems*, 6(4):749–760, 2019. 2

[5] Zhenyu Huang, Joey Tianyi Zhou, Xi Peng, Changqing Zhang, Hongyuan Zhu, and Jiancheng Lv. Multi-view Spectral Clustering Network. In *IJCAI*, 2019. 2, 3

[6] Robert Jenssen, Jose C. Principe, Deniz Erdogmus, and Torbjørn Eltoft. The Cauchy–Schwarz divergence and Parzen windowing: Connections to graph theory and Mercer kernels. *Journal of the Franklin Institute*, 343(6):614–629, 2006. 3

[7] Michael Kampffmeyer, Sigurd Løkse, Filippo M. Bianchi, Lorenzo Livi, Arnt-Børre Salberg, and Robert Jenssen. Deep Divergence-based Approach to Clustering. *Neural Networks*, 113:91–101, 2019. 2, 3

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 4

[9] Z. Li, Q. Wang, Z. Tao, Q. Gao, and Z. Yang. Deep Adversarial Multi-view Clustering Network. In *IJCAI*, 2019. 2

[10] Yiqiao Mao, Xiaoqiang Yan, Qiang Guo, and Yangdong Ye. Deep Mutual Information Maximin for Cross-Modal Clustering. In *IJCAI*, 2021. 2

[11] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical report, 1996. 4

[12] Xiukun Sun, Miaomiao Cheng, Chen Min, and Liping Jing. Self-Supervised Deep Multi-View Subspace Clustering. In *ACML*, 2019. 2

[13] Xiaoliang Tang, Xuan Tang, Wanli Wang, Li Fang, and Xian Wei. Deep Multi-view Sparse Subspace Clustering. In *ICNCC*, 2018. 2

[14] Daniel J. Trosten, Sigurd Løkse, Robert Jenssen, and Michael Kampffmeyer. Reconsidering Representation Alignment for Multi-view Clustering. In *CVPR*, 2021. 2, 3

[15] Qianqian Wang, Zhiqiang Tao, Wei Xia, Quanxue Gao, Xiaochun Cao, and Licheng Jiao. Adversarial Multiview Clustering Networks With Adaptive Fusion. *IEEE transactions on neural networks and learning systems*, PP, 2022. 2

[16] Shiye Wang, Changsheng Li, Yanming Li, Ye Yuan, and Guoren Wang. Self-Supervised Information Bottleneck for Deep Multi-View Subspace Clustering. *arXiv:2204.12496 [cs]*, 2022. 2

[17] Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On Deep Multi-View Representation Learning. In *ICML*, 2015. 2

[18] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747 [cs, stat]*, 2017. 4

[19] Bowen Xin, Shan Zeng, and Xiuying Wang. Self-Supervised Deep Correlational Multi-View Clustering. In *IJCNN*, 2021. 2

[20] Jie Xu, Yazhou Ren, Guofeng Li, Lili Pan, Ce Zhu, and Zenglin Xu. Deep embedded multi-view clustering with collaborative training. *Information Sciences*, 573:279–290, 2021. 2

[21] Jie Xu, Yazhou Ren, Huayi Tang, Xiaorong Pu, Xiaofeng Zhu, Ming Zeng, and Lifang He. Multi-VAE: Learning Disentangled View-Common and View-Peculiar Visual Representations for Multi-View Clustering. In *ICCV*, 2021. 2

[22] Ming Yin, Weitian Huang, and Junbin Gao. Shared Generative Latent Representation Learning for Multi-View Clustering. In *AAAI*, 2020. 2

[23] Xianchao Zhang, Jie Mu, Linlin Zong, and Xiaochun Yang. End-To-End Deep Multimodal Clustering. In *ICME*, 2020. 2

[24] Xianchao Zhang, Xiaorui Tang, Linlin Zong, Xinyue Liu, and Jie Mu. Deep Multimodal Clustering with Cross Reconstruction. In *PAKDD*, 2020. 2

[25] Runwu Zhou and Yi-Dong Shen. End-to-End Adversarial-Attention Network for Multi-Modal Clustering. In *CVPR*, 2020. 2, 5

[26] Pengfei Zhu, Binyuan Hui, Changqing Zhang, Dawei Du, Longyin Wen, and Qinghua Hu. Multi-view Deep Subspace Clustering Networks. *arXiv:1908.01978 [cs]*, 2019. 2

(a) Weight of reconstruction loss (w^{SV}).

	NoisyMNIST				Caltech7			
Rec. weight	0.01	0.1	1.0	10.0	0.01	0.1	1.0	10.0
AE–DDC	1.00 (0.03)	0.94 (0.03)	1.00 (0.03)	0.94 (0.01)	0.41 (0.01)	0.41 (0.03)	0.44 (0.02)	0.45 (0.02)
AECoDDC	0.99 (0.00)	0.99 (0.00)	0.99 (0.00)	0.99 (0.00)	0.40 (0.05)	0.33 (0.02)	0.34 (0.03)	0.49 (0.04)
AECoKM	0.74 (0.02)	0.70 (0.04)	0.74 (0.03)	0.93 (0.02)	0.07 (0.01)	0.05 (0.00)	0.04 (0.01)	0.04 (0.02)

(b) Weight of contrastive loss (w^{MV}).

	NoisyMNIST				Caltech7			
Con. weight	0.01	0.1	1.0	10.0	0.01	0.1	1.0	10.0
AECoDDC	1.00 (0.00)	0.99 (0.00)	0.99 (0.00)	0.99 (0.00)	0.46 (0.02)	0.40 (0.05)	0.19 (0.03)	0.09 (0.01)
AECoKM	0.89 (0.01)	0.73 (0.03)	0.77 (0.01)	0.67 (0.02)	0.04 (0.02)	0.04 (0.01)	0.06 (0.01)	0.05 (0.00)

(c) Temperature in the contrastive loss (τ).

	NoisyMNIST				Caltech7			
τ	0.01	0.07	0.1	1.0	0.01	0.07	0.1	1.0
AECoDDC	0.99 (0.00)	1.00 (0.00)	0.99 (0.00)	1.00 (0.00)	0.31 (0.03)	0.39 (0.01)	0.35 (0.02)	0.48 (0.01)
AECoKM	0.99 (0.00)	0.91 (0.02)	0.74 (0.02)	0.78 (0.05)	0.34 (0.01)	0.05 (0.01)	0.06 (0.01)	0.46 (0.01)

(d) Weight of the entropy regularization (λ).

	NoisyMNIST				Caltech7			
λ	0.5	1.5	5.0	10.0	0.5	1.5	5.0	10.0
MV-IIC	0.03 (0.01)	0.81 (0.01)	0.82 (0.00)	0.82 (0.00)	0.04 (0.01)	0.64 (0.04)	0.60 (0.01)	0.52 (0.01)
InfoDDC	0.21 (0.02)	0.37 (0.02)	0.84 (0.04)	0.94 (0.07)	0.60 (0.06)	0.60 (0.02)	0.57 (0.01)	0.51 (0.01)

Table 8. Results (NMI) of hyperparameter sweeps for the new instances.

[27] Linlin Zong, Faqiang Miao, Xianchao Zhang, and Bo Xu. Multimodal Clustering via Deep Commonness and Uniqueness Mining. In *ICIKM*, 2020. 2