This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

MetaViewer: Towards A Unified Multi-View Representation

Ren Wang
Shandong University
xxlifelover@gmail.com

Haoliang Sun*
Shandong University
haolsun.cn@gmail.com

Yuling Ma
Shandong Jianzhu University
mayuling20@sdjzu.edu.cn

Xiaoming Xi
Shandong Jianzhu University
fyzq10@126.com

Yilong Yin*
Shandong University
ylyin@sdu.edu.cn

Abstract

Existing multi-view representation learning methods typically follow a specific-to-uniform pipeline, extracting latent features from each view and then fusing or aligning them to obtain the unified object representation. However, the manually pre-specified fusion functions and aligning criteria could potentially degrade the quality of the derived representation. To overcome them, we propose a novel uniform-to-specific multi-view learning framework from a meta-learning perspective, where the unified representation no longer involves manual manipulation but is automatically derived from a meta-learner named MetaViewer. Specifically, we formulated the extraction and fusion of view-specific latent features as a nested optimization problem and solved it by using a bi-level optimization scheme. In this way, MetaViewer automatically fuses view-specific features into a unified one and learns the optimal fusion scheme by observing reconstruction processes from the unified to the specific over all views. Extensive experimental results in downstream classification and clustering tasks demonstrate the efficiency and effectiveness of the proposed method.

1. Introduction

Multi-view representation learning aims to learn a unified representation of the entity from its multiple observable views for the benefit of downstream tasks [35, 43, 58]. Each view acquired by different sensors or sources contains both view-shared consistency information and view-specific information [8]. The view-specific part further consists of complementary and redundant components, where the former can be considered as a supplement to the consistency information, while the latter is view-private and may be adverse for the unified representation [16]. Therefore, a high-quality representation is required to retain the consistency and complementary information, as well as filter out the view-private redundant ones [51].

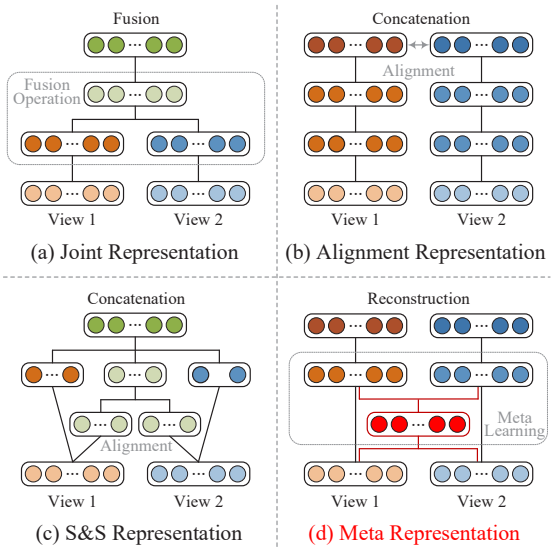


Figure 1. (a), (b) and (c) show three multi-view learning frameworks following the *specific-to-uniform* pipeline, where the unified representation is obtained by fusing or concatenating view-specific features. (d) illustrates our *uniform-to-specific* manner, where a meta-learner learns to fusion by observing reconstruction from unified representation to specific views.

Given the data containing two views, x_1 and x_2, prevailing multi-view learning methods typically follow a *specific-to-uniform* pipeline and can be roughly characterized as:

$$H := f(x_1; W_f) \circ g(x_2; W_g), \tag{1}$$

where f and g are encoding (or embedding [27]) functions that map the original view data into the corresponding latent

*Corresponding authors

features with the trainable parameters W_f and W_g. These latent features are subsequently aggregated into the unified representation H using the designed aggregation operator $\circ$. With different aggregation strategies, existing approaches can be further subdivided into the joint, alignment, and a combined share-specific (S&S) representation [22, 26].

Fig. 1 (a) - (c) show the above three branches of the *specific-to-uniform* framework. Joint representation focuses on the integration of complementary information by directly fusing latent features, where the $\circ$ is represented as fusion strategies, such as graph-based modules [33], neural networks [10], or other elaborate functions [7, 34]. While alignment representation seeks alignment between view-specific features to retain consistency information and specifies $\circ$ as the alignment operator, measured by distance [11], correlation [24, 47], or similarity [25, 38, 39]. The aligned features can be concatenated as the unified representation for downstream tasks. As a trade-off strategy, S&S representation explicitly distinguishes latent features into shared and specific representations and only aligns the shared part [20, 30, 53].

Despite demonstrating promising results, the *specific-to-uniform* framework inherently suffers from potential risks in the following two aspects: (1) Compared with the data-oriented fusion rules, manually pre-specified rules are designed to compute the unified representation by concatenating or fusing latent features, restricting its extensibility in complex real-world applications. (2) Even if we can find a well-performing fusion scheme, the sequential training manner limits the model to constrain the various component information separately. For example, it is difficult to automatically separate out view-private redundant information from the feature level [22, 37]. Recent researches have attempted to address the second issue by decomposing latent features using matrix factorization [61] or hierarchical feature modeling [51], but still cannot avoid manually pre-specified fusion or even decomposition strategies.

In this work, we provide a new meta-learning perspective for multi-view representation learning and propose a novel *uniform-to-specific* framework to address these potential risks. Fig. 1 (d) shows the overall schematic. In contrast to the *specific-to-uniform* pipeline, the unified representation no longer involves manual manipulation but is automatically derived from a meta-learner named MetaViewer. To train the MetaViewer, we first decouple the learning of view-specific latent features and unified meta representation. These two learning processes can then be formulated as a nested optimization problem and eventually solved by a bi-level optimization scheme. In detail, MetaViewer fuses view-specific features into a unified one at the outer level and learns the optimal fusion scheme by observing reconstruction processes from the unified to the specific over all views at the inner level. In addition, our *uniform-to-specific* framework is compatible with most existing objective functions and pre-text tasks to cope with complex real-world scenarios. Extensive experiments validate that MetaViewer achieves comparable performance to the state-of-the-art methods in downstream clustering and classification tasks. The core contributions of this work are as follows.

1. We provide a new meta-learning perspective and develop a novel *uniform-to-specific* framework to learn a unified multi-view representation. To the best of our knowledge, it could be the first meta-learning-based work in multi-view representation learning community.

2. We propose MetaViewer, a meta-learner that formulates the modeling of view-specific features and unified representation as a nested bi-level optimization and ultimately meta-learns a data-driven optimal fusion.

3. Extensive experiments on multiple benchmarks validate that our MetaViewer achieves comparable performance to the existing methods in two downstream tasks.

2. Related Work

2.1. Multi-view representation learning

Representation learning is not a new topic and plays a critical role in numerous downstream tasks [29, 36, 49]. This work focuses on multi-view representation in unsupervised deep learning scope [56]. Related studies can be roughly summarized into two categories [54]. One is the deep extension of traditional methods, where representative ones include deep canonical correlation analysis (DCCA) and its variants [44, 57]. DCCA [2] intends to discover the nonlinear mapping for two views to a common space in which their correlations are maximally preserved. These methods benefit from a sound theoretical foundation, but also usually have strict restrictions on the number and form of views.

Another alternative is deep multi-view learning [5, 32]. Early deep-based approaches attempted to design effective architectures for multiple views, such as CNN-based [12, 42] and GAN-based models [18, 52]. Recent works resort to leveraging mutual or comparative information to derive the constraint for parameters [3, 27, 55]. Most of them follow the pipeline from view-specific features to unified representation. In contrast, our MetaViewer learns by observing the learning from the uniform to the specific. The most related work is AE^2-Nets [60], which treats the unified representation as a set of trainable parameters that are optimized during the degradation process of each view. The essential difference is that we learn the data-oriented fusion strategy rather than optimizing the unified representation itself.

2.2. Meta-learning

Optimization-based meta-learning is a classic application of bi-level optimization designed to learn task-level knowledge to quickly handle new tasks [19, 21]. A typical work,

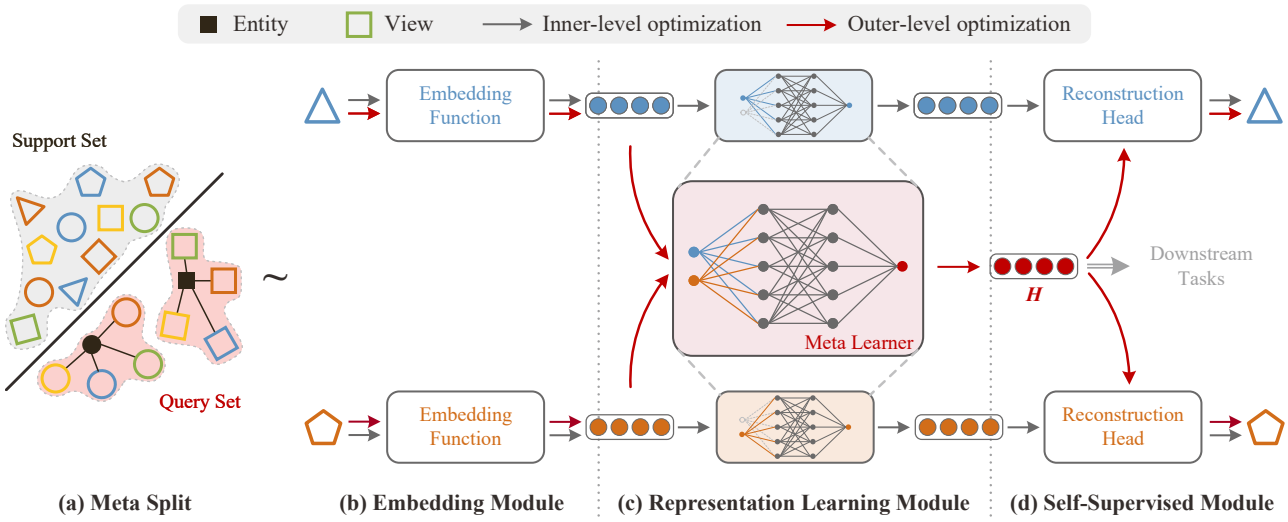


Figure 2. The overall framework of MetaViewer, contains (a) the meta-split of multi-view data and three modules: (b) embedding module, (c) representation module, and (d) self-supervised module. These modules are trained with a bi-level optimization. Inner-level (dark gray arrows) learns the view-specific reconstruction on the support set, and outer-level (red arrows) updates the entire model to learn the fusion scheme and the unified representation H for downstream tasks by validating on the query set.

MAML [13], learns a set of initialization parameters to solve different tasks with a few steps of updates. Similar meta paradigm has been used to learn other manually designed parts, such as the network structure [28], optimizer [62], and even sample weight [40, 41, 48]. Similarly, we try to meta-learn the fusion rule of multi-view features for a unified representation. There also exist some works that consider both meta-learning paradigms and multi-view data [14, 31, 45]. However, they are dedicated to exploiting the rich information contained in multiple views to improve the performance of the meta-learner in a few-shot or self-supervised scenario. Instead, we train a meta-learner to derive high-quality shared representations from multi-view data. To the best of our knowledge, this could be the first work to learn multi-view representation with a meta-learning paradigm.

3. MetaViewer

Given a set of unlabeled multi-view data, $D = \{x^i \in \mathbb{R}^{d_x}\}_{i=1}^N$, where N and superscript i are the number and index of entity samples, respectively. Each sample $x^i = \{x_1^i, x_2^i, \ldots, x_v^i\}_{v=1}^V$ contains V views, where subscript v is the index of views. Our goal is to learn a unified, high-quality representation H for each entity by observing these views and filtering view-private information as much as possible. The overall framework of our MetaViewer is shown in Fig. 2, including three main modules and a bi-level optimization process. The outer-level trains a meta-learner to learn an optimal fusion function and derive the unified representation H, and the inner-level reconstructs original views from H in a few update steps, which explicitly models and separates view-private information and ensure the quality of the representation. In the following subsections, we first introduce the entire structure of the MetaViewer and then elaborate on the bi-level optimization process.

3.1. The entire structure

Embedding module aims to transform heterogeneous views into the latent feature space, where transformed view embeddings have the same dimension as each other. To this end, we conduct a view-specific embedding function f_v for each view, where $v = 1, 2, \ldots, V$. Given the v-th view data x_v of the entity x, the corresponding embedding $z_v \in \mathbb{R}^d$ can be computed by

$$z_v = f_v(x_v, \theta_{f_v}), \tag{2}$$

where f_v is typically instantiated as a multi-layer neural network with learnable parameters θ_{f_v}.

Representation learning module maps the obtained embedding to the view-specific or unified representation by constructing a meta-learner m (i.e., MetaViewer). Note that the two types of representations are both derived from the same set of parameters, but at different learning stages (see 3.2). The view-specific representation is obtained at the inner level, where MetaViewer receives the embedding of the v-th view and outputs the corresponding representation h_v. In contrast, the unified representation H is learned by MetaViewer from all view embeddings at the outer level and is ultimately used for downstream tasks. Therefore, the structure of MetaViewer should be flexible enough to meet the two requirements above simultaneously. Specifically,

MetaViewer is implemented as a channel-oriented 1-d convolutional layer (C-Conv) with a non-linear function (e.g., ReLU [17]), as shown in Fig. 2 (c).

On the one hand, at the outer level, we first concatenate all view embeddings at the channel level, i.e., the number of the channel in the concatenated embedding is equal to the number of views, and then train the MetaViewer to learn the fusion of cross-view information $H \in \mathbb{R}^{d_h}$ by

$$H = m(z_{cat}, \omega), \tag{3}$$

where $z_{cat} \in \mathbb{R}^{d \times V}$ indicates the concatenated embedding and ω is the parameter of MetaViewer. On the other hand, the inner-level initializes and trains MetaViewer to learning the representation $h_v \in \mathbb{R}^{d_h}$ of the v-th view via

$$h_v = m_v(z_v, \omega_v), \tag{4}$$

where m_v denotes the learner with the learnable parameter ω_v for handling the v-th view in the inner level, also known as the base learner to avoid confusion. It is worth noting that ω_v is initialized by a subset of the MetaViewer parameters.

Self-supervised module provides effectively supervised information for model training by constructing a series of pre-text tasks[1]. These tasks serve as the heads of our framework. Given the input z, the output s can be formalized as

$$s = t(z, \theta_t), \tag{5}$$

where t and θ_t are the network and its parameters involved in pre-text tasks, respectively. Taking the reconstruction head as an example, it achieves the reconstruction object by re-mapping the representation back to the original view space. In this case, the $s := \hat{x}_v$ indicates the reconstruction result for input x_v, and $t(\cdot, \theta_t) := t_v(\cdot, \theta_{t_v})$ is a multi-layer neural network with the opposite structure to the embedding module. More similar pre-text tasks have been studied in previous works [55, 57], and this is not the focus of this work.

3.2. Training via bi-level optimization

We now have the entire structure, which can be end-to-end trained following the proposed *uniform-to-specific* pipeline to derive the unified multi-view representation. The training flowchart and step are shown in the top row in Fig. 3, where the learning of view-private features and unified representation are formulated as a nested bi-level optimization problem. Inner-level focus on independent training on a specific view, which receives a given representation and reconstructs it to each view. Outer-level updates the meta-learner to find the optimal fusion rule through observing the training processes across all views. This pipeline decouples the unified entity representations from view-specific features and enables data-driven fusion. Before the detailed description, we introduce a split strategy for multi-view data in meta-learning style to adapt the bi-level training.

Meta-split of multi-view data. Consider a batch multi-view entity samples $\{D_v^{batch}\}_{v=1}^V$ from set D. We randomly and proportionally divide it into two disjoint subsets, marked as support set $S = \{S_v\}_{v=1}^V$ and query set $Q = \{Q_v\}_{v=1}^V$, respectively. As shown in 2 (a), support set is used in inner-level for leaning view-specific information, so the sample attributes in it could be ignored. In contrast, query set retains both view and sample attributes for outer-level optimization.

Inner-level optimization. Without loss of generality, take the inner-level update after the o-th outer-level optimization as an example. Let ω^o be the lasted parameters of the MetaViewer, and $\theta_v^o = \{\theta_{f_v}^o, \theta_{t_v}^o\}$ denotes the lasted parameters in embedding and self-supervised modules for brevity. We first initial base learner from meta-learner, i.e., $\omega_v^0 = \omega^o$, and make a copy of θ_v^o for $\tilde{\theta}_v$. Note that the *copy* means gradients with respect to the θ_v^o will not be back-propagated to $\tilde{\theta}_v$ and vice versa. Thus, ω_v^0 and $\tilde{\theta}_v$ form the initial states of parameters in the inner-level optimization. Suppose $\mathcal{L}_v^{in}$ is the loss function of the inner-level with respect to the v-th view, the corresponding update goal is

$$\omega_v^*(\omega^o) = \arg\min \mathcal{L}_v^{in}\left(\omega_v(\omega^o), \tilde{\phi}_v; S_v\right). \tag{6}$$

Consider a gradient descent strategy (e.g., SGD [4]), we can further write the update process of ω_v:

$$\omega_v^i = \omega_v^{i-1} - \beta \frac{\partial \mathcal{L}_v^{in}}{\partial \omega_v^{i-1}}, \ldots, \omega_v^0 = \omega^o, \tag{7}$$

where β and i denote the learning rate and iterative step of inner-level optimization, respectively.

Outer-level optimization. After several inner-level updates, we obtain a set of optimal view-specific parameters for the support set. Outer-level then updates the meta-learner, embedding, and self-supervised modules by training on the query set. With the loss function $\mathcal{L}^{out}$, the outer-level optimization goal is

$$\omega^*, \{\theta_v^*\}_{v=1}^V = \arg\min \mathcal{L}^{out}\left(\{\omega_v^*(\omega), \phi_v\}_{v=1}^V; Q\right). \tag{8}$$

By alternately optimizing Eq. 6 and Eq. 8, we end up with the optimal meta-parameters ω^* and a set of view-specific parameters $\{\theta_v^*\}_{v=1}^V$. Given a test sample x_{test}, the corresponding unified representation is derived by feeding it sequentially into the embedding module and the meta-learner. The overall framework of MetaViewer is shown in Alg. 1.

3.3. Specific-to-uniform versus uniform-to-specific

We discuss the difference between *specific-to-uniform* and our *uniform-to-specific* pipeline through three aspects shown in Fig. 3, including the overall flowchart, update steps,

[1]Constructing pre-text tasks is a classic strategy in self-supervised learning [1], which emphasizes the use of intrinsic properties within or between data as ground-truth to provide supervised information.

Algorithm 1 The framework of our MetaViewer.

Require: Training dataset D, meta parameters ω, base parameters $\{\omega_v\}_{v=1}^V$, view-specific parameters $\{\theta_v\}_{v=1}^V$, the number of views V, the iteration step in inner-level optimization T.

Initialize ω, $\{\theta_v\}_{v=1}^V$;
while not done **do**
  # *Outer-level*
  Sample and meta-split a batch set from D:
  $\{D_v^{batch}\}_{v=1}^V = \{S_v\}_{v=1}^V + \{Q_v\}_{v=1}^V$.
  for $t = 1, \ldots, T$ **do**
    for $v = 1, \ldots, V$ **do**
      # *Inner-level*
      Initialize $\omega_v = \omega$, $\tilde{\theta_v} = \theta_v$;
      Optimize $\omega_v(\omega)$ and $\tilde{\theta_v}$ via Eq. 6.
    end for
  end for
  Optimize ω and $\{\phi_v\}_{v=1}^V$ via Eq. 8.
end while

and key gradients. For clarity, we here slightly confuse the notation and let θ_v and $\mathcal{L}_v^{in}$ correspond to the view-specific parameters and loss functions in *specific-to-uniform* pipeline, respectively. Similarly, ω and $\mathcal{L}^{out}$ denote the view-shared parameters and loss functions, respectively. Let us start by reviewing two branches in *specific-to-uniform* framework.

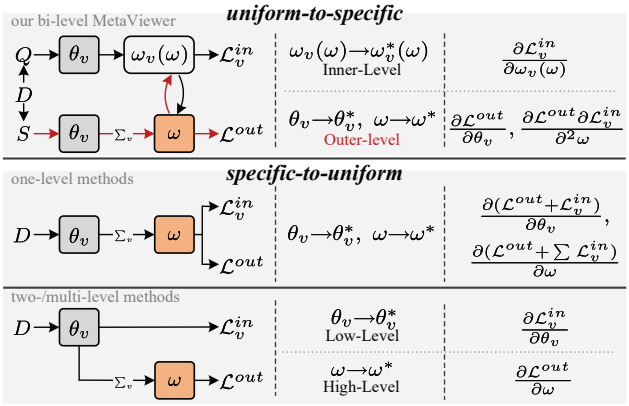
One-level methods is a classical branch in *specific-to-uniform* framework, where parameters $\{\theta_v\}_{v=1}^V$ and ω are simultaneously optimized by combining the functions $\mathcal{L}_v^{in}$ and $\mathcal{L}^{out}$.

Multi-level methods also follow the *specific-to-uniform* framework, the difference is that they hierarchically learn specific and shared features to avoid interference. That is, parameters θ_v are updated at the low level by minimizing the $\mathcal{L}_v^{in}$, and parameters ω are updated at the high level by minimizing the $\mathcal{L}^{out}$, respectively.

Bi-level MetaViewer further decouples view-private information, except for view-specific and view-sharing components. Consider the simplest case, where the base learner updates only one step on each view in the inner loop, i.e., $\omega_v^* = \omega - \beta\nabla_\omega \mathcal{L}_v^{in}(\omega)$. Thus, the final optimization of ω in the outer level in Eq. 8 can be rewritten as

$$\min_{\omega} \sum_v \mathcal{L}^{out}(\omega_v^*) = \sum_v \mathcal{L}^{out}\left(\omega - \beta\nabla_\omega \mathcal{L}_v^{in}(\omega)\right). \quad (9)$$

Intuitively, the optimal ω is expected to be close enough to each view that the view-specific representation can be produced after once updating. In other words, our *uniform-to-specific* framework requires explicit modeling the gap caused by view-private information when mapping from a unified representation to a particular view, which is achieved by inner-level optimization in MetaViewer.



(a) Flowcharts (b) Steps (c) Gradients

Figure 3. Comparison of two frameworks in terms of flowchart, update steps, and key gradients. The top row is our bi-level MetaViewer with *uniform-to-specific* framework, where red and black arrows indicate the flow at outer-level and inner-level, respectively. The bottom two rows show two main branches in *uniform-to-specific* framework, i.e., one-level methods [47] and two- or multi-level methods [51].

3.4. The instances of the objective function

Our *uniform-to-specific* framework emphasizes learning from the training process in the inner loop. In unsupervised multi-view learning, this process typically refers to the reconstruction on views, thus the $\mathcal{L}_v^{in}$ is specified as the reconstruction loss [27, 60]

$$\mathcal{L}_v^{in} = \mathcal{L}_v^{rec}(S_v, S_v^{rec}) = \|S_v - S_v^{rec}\|_F^2. \quad (10)$$

While parameters updated at the outer level can be constrained by richer self-supervised feedbacks, as maintained in Sec. 3.1. Here we provide two instances of the outer-level loss function to demonstrate how MetaViewer can be extended with different learning objectives.

MVer-R adopts the same reconstruction loss as the inner-level, and $\mathcal{L}^{out} = \sum_v \mathcal{L}_v^{rec}(Q_v, Q_v^{rec})$, which is the purest implementation of MetaViewer.

MVer-C additionally utilizes a contrastive objective, where the similarities of views belonging to the same entity (i.e., positive pairs) should be maximized and those of different entities (i.e., negative pairs) should be minimized, i.e., $\mathcal{L}^{out} = \sum_v \left(\mathcal{L}_v^{rec} + \sum_{v' , v' \neq v} \mathcal{L}_{v,v'}^{con}\right)$. Following previous work [18, 51, 55], the contrastive loss $\mathcal{L}_{v,v'}^{con}$ is formed as

$$\mathcal{L}_{v,v'}^{con} = -\frac{1}{N_Q}\sum_{i=1}^{N_Q} log \frac{e^{d(q_v^i, q_{v'}^i)/\tau}}{\sum_{j=1, j\neq i}^{N_Q} e^{d(q_v^i, q_v^j)/\tau} + \sum_{j=1}^{N_Q} e^{d(q_v^i, q_{v'}^j)/\tau}}, \quad (11)$$

where q_v^i is the v-th view of the i-th query sample and d is the similarity metric (e.g., cosine similarity [6]). The N_Q and τ denote the number of query set samples and the

Datasets	#Views	#Classes	#Samples (train, val, test)	View Dimensions
Handwirtten	2	10	2000 (1200, 400, 400)	240; 216
RGB-D	2	50	500 (300, 100, 100)	12288; 4096
Animal	2	50	10158 (6074, 2011, 2073)	4096; 4096
Fashion-MV	3	10	10000 (6000, 2000, 2000)	784; 784; 784
Caltech101-20	6	20	2386 (1425, 469, 492)	48; 40; 254; 1984; 512; 928

Table 1. The attributes for all datasets used in our experiments.

Methods	Handwritten			RGB-D			Animal			Fashion-MV			Caltech101-20		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
Baseline	43.55	52.19	25.68	48.30	80.97	04.95	45.79	66.22	33.87	50.91	45.98	32.09	36.63	48.96	23.54
DCCA [2] (2013)	57.25	69.80	52.15	51.00	82.99	52.02	60.34	68.51	46.10	70.70	80.42	61.80	38.62	50.88	22.73
DCCAE [47] (2015)	63.00	75.04	59.29	48.00	81.58	48.34	65.17	66.82	48.83	71.05	81.12	62.34	36.59	52.24	25.25
MIB [8] (2020)	63.25	67.58	52.16	50.00	81.13	51.27	60.89	61.98	52.49	57.20	73.83	47.62	35.98	47.00	22.18
WTNNM [15] (2020)	75.69	77.62	61.51	52.54	81.95	52.46	71.92	71.43	61.06	83.33	86.30	77.99	39.74	55.58	26.92
TLRR [23] (2021)	78.60	77.99	66.40	52.87	82.38	53.37	69.79	73.01	61.23	79.54	84.01	78.54	40.18	52.38	27.60
MFLVC [51] (2022)	64.00	64.53	48.85	53.00	83.31	54.07	74.10	76.08	64.28	83.20	88.75	78.93	36.59	58.36	26.87
DCP [27] (2022)	66.25	70.56	56.10	52.00	82.04	52.64	72.77	74.02	65.48	62.60	68.38	54.30	36.79	44.37	23.50
MVer-R (ours)	75.00	78.53	67.21	53.00	82.41	53.04	**76.49**	**78.25**	65.28	80.80	88.13	75.05	41.87	58.52	29.19
MVer-C (ours)	**86.25**	**78.96**	**72.25**	**57.00**	**84.97**	**57.07**	75.92	78.01	**66.07**	**85.40**	**88.76**	**80.07**	**45.12**	**60.86**	**35.00**

Table 2. Clustering results of all methods on six datasets. Bold and underline denote the best and second-best results, respectively.

temperature parameter, respectively. Note that, the derived meta representation can also be used in contrastive learning as an additional novel view.

4. Experiments

In this section, we present extensive experimental results to validate the quality of the unified representation derived from our MetaViewer. The remainder of the experiments are organized as follows: Subsection 4.1 lists datasets, compared methods, and implementation details. Subsection 4.2 compares the performance of our method with classical and state-of-the-art methods on two common downstream scenarios, clustering and classification tasks. Comparisons with manually designed fusion and ablation studies are shown in Subsections 4.3 and 4.4, respectively.

4.1. Experimental Setup

Datasets. To comprehensively evaluate the effectiveness of our MetaViewer, we conduct five multi-view benchmarks in experiments. All datasets are scaled to $[0, 1]$ and split into training, validation, and test sets in the ratio of $6:2:2$, as shown in Tab. 1. **Handwritten** contains $2,000$ handwritten digital images from 0 to 9, where two types of descriptors, i.e., 240-D pixel average in 2×3 windows and 216-D profile correlations, are selected as two views [60]. **RGB-D** dataset contains visual and depth images of 300 distinct objects across 50 categories [46, 63]. Two views are obtained by flattening the $64 \times 64 \times 3$ color images and 64×64 depth images. **Animal** consists of 10158 images from 50 classes with two views. Unlike other datasets, these two views are features extracted by deep networks [59]. **Fashion-MV** is an image dataset that contains 10 categories with a total of $30,000$ fashion products. It has three views and each of which consists of 10,000 gray images sampled from the same category [50]. **Caltech101-20** is a subset of the Caltech101 image set [9], which consists of $2,386$ images of 20 subjects. Six features are used, including Gabor, Wavelet Moments, CENTRIST, HOG, GIST, and LBP.

Compared methods. We compare the performance of MetaViewer with seven representative multi-view learning methods, including two classical methods (DCCA [2] and DCCAE [47]) and five state-of-the-art methods (WTNNM [15], TLRR [23], MIB [8], MFLVC [51] and DCP [27]). Among them, DCCA and DCCAE are the deep extensions of traditional correlation strategies. WTNNM and TLRR are representative tensor-based methods. MIB is a typical generative method with mutual information constraints. DCP learns unified representation in both unsupervised and supervised scenarios, and we report the unsupervised version for a fair comparison. In particular, MFLVC also notices the view-private redundant information and designs a multi-level feature network for clustering tasks. In addition, we also compare the model without MetaViewer as a baseline, where the unified representation is obtained by concatenating view-specific features output by the embedding module.

Implementation details. For a fair comparison, all methods are trained from scratch, and all deep-network-based

Methods	Handwritten			RGB-D			Animal			Fashion-MV			Caltech101-20		
	ACC	Prec.	F-score	ACC	Prec.	F-score	ACC	Prec.	F-score	ACC	Prec.	F-score	ACC	Prec.	F-score
Baseline	87.00	87.32	87.04	14.00	06.45	07.70	71.78	65.02	63.10	87.50	87.63	87.54	75.41	51.95	46.21
DCCA [2] (2013)	88.25	89.20	88.05	30.00	21.10	22.04	62.84	61.61	60.65	84.90	85.22	83.54	71.54	45.27	39.81
DCCAE [47] (2015)	90.00	90.48	89.92	24.00	16.00	16.91	66.82	62.53	62.83	85.35	85.97	83.84	71.54	60.57	43.25
MIB [8] (2020)	79.00	83.90	78.52	33.00	28.50	27.37	59.32	63.78	62.13	86.80	86.80	86.55	72.72	61.64	52.47
WTNNM [15] (2020)	96.77	96.20	96.36	45.00	47.18	43.90	69.90	67.40	65.48	94.50	94.50	94.58	83.27	80.49	74.78
TLRR [23] (2021)	97.00	97.05	97.17	49.00	53.32	47.66	71.90	69.52	68.35	96.35	96.28	96.40	85.55	77.30	75.24
MFLVC [51] (2022)	94.00	94.20	94.01	44.00	46.09	41.81	75.62	72.17	70.81	96.50	96.52	96.49	85.37	71.83	69.07
DCP [27] (2022)	97.25	97.30	97.24	37.00	28.87	30.78	77.95	73.43	70.01	89.25	82.06	82.90	**92.48**	89.41	84.58
MVer-R (ours)	97.00	97.08	97.00	51.00	53.65	48.73	77.69	73.91	71.22	96.85	96.37	96.48	92.28	89.46	84.21
MVer-C (ours)	**97.75**	**97.90**	**97.75**	**56.00**	**55.20**	**52.78**	**78.03**	**74.56**	**71.55**	**97.70**	**96.78**	**97.07**	92.16	**90.68**	**85.72**

Table 3. Classification results of all methods on six datasets. Bold and underline denote the best and second-best results, respectively.

methods share the same embedding backbone, following the previous work [51]. For two-view-based methods, such as DCCA, DCCAE, and MIB, we report the results of the two best-performing views. The final unified representation H in alignment-based methods is a concatenation of the processed features, and the representation dimension is specified as $d_h = 256$. The results on clustering and classification tasks are obtained by subsequent K-means and SVM classifier, respectively. For MetaViewer, we train $2,000$ epochs for all benchmarks, and set the batch size is 32 for RGBD and 128 for others. The learning rates in outer- and inner-level are set to 10^{-4} and 10^{-3}, respectively. We use a single convolutional layer with 32 kernels of size 5 as the meta-learner, and we set the proportion of support set and the update step in the inner level to 0.2 and 2, respectively. All experiments have been verified using the PyTorch library on a single RTX3090. Code is available at https://github.com/xxLifeLover/MetaViewer.

4.2. Performance on downstream tasks

Clustering results. Tab. 2 lists the results of the clustering task, where the performance is measured by three standard evaluation metrics, i.e., Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI). A higher value of these metrics indicates better clustering performance. It can be observed that (1) our MVer-C variant significantly outperforms other compared methods on all benchmarks; (2) the second-best results appear between MVer-R and MFLVC, both of which explicitly separate the view-private information; (3) a larger number of categories and views is the main reason for the degradation of clustering performance, and our Metaviewer improves most significantly in such a scenario (e.g., Fashion-MV and Caltech101-20).

Classification results. Tab. 3 lists the results of the classification task, where three common metrics are used, including Accuracy, Precision, and F-score. A higher value indicates better classification performance. Similar to the clustering results, two variants of MetaViewer significantly outperform the comparison methods. It is worth noting that (1) DCP learns unified representation and therefore achieves the second-best result instead of MFLVC. (2) The number of categories is the main factor affecting the classification performance, and our method obtains the most significant improvement in the RGB-D dataset with up to 50 classes.

Strategies	Rules	ACC↑	NMI↑	ARI↑	MSE↓
Sum	$z^x + z^y$	69.25	71.89	59.02	1.84
Max	$max(z^x, z^y)$	80.75	73.93	63.75	-
Concat.	$cat[z^x, z^y]$	78.75	72.02	61.52	**1.77**
Linear	$l(z^x, z^y, \theta_l)$	85.00	77.40	69.71	4.74
C-Conv	$m(z^x, z^y, \omega)$	69.75	65.21	51.33	2.37
MetaViewer	*meta-learning*	**86.25**	**78.96**	**72.25**	2.45

Table 4. Clustering resulting on the Handwritten dataset.

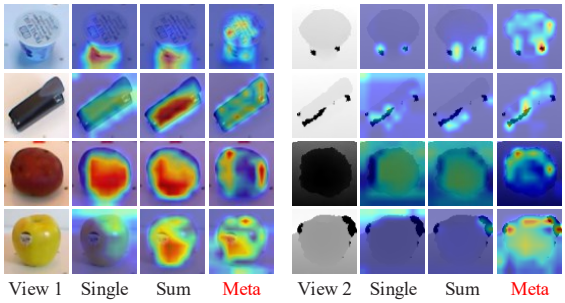
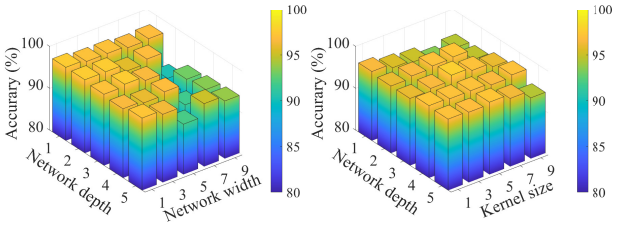


Figure 4. Class activate maps on RGB-D samples for different fusion strategies, including single views, Sum, and our MetaViewer. View 1 and view 2 are the RGB and depth view, respectively.

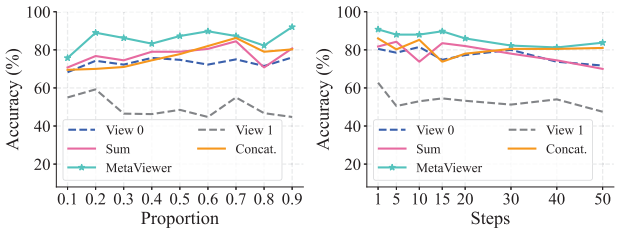
4.3. Comparison with manually designed fusion.

As mentioned in 3.3, MetaViewer essentially learns an optimal fusion rule that preserves as much consistency and complementary information as possible while filtering out view-private information. To verify this, we compare it with commonly used fusion strategies [26], including *sum*, *maxima*, *concatenation*, *linear layer* and *C-Conv*. The former three are the specified fusion rules without trainable param-



(a) Effect of network width (b) Effect of kernel size

Figure 5. Effect of meta-learner architectures with different depth, width, and kernel size on classification accuracy.



(a) Proportion of support set (b) Steps of inner-level update

Figure 6. Effect of (a) different meta-division ratios and (b) the number of inner-loop iterations on classification accuracy.

eters, and the remaining two are the trainable fusion layer trained via the *specific-to-uniform* manner. Tab. 4 lists the clustering results and an additional MSE score on the Handwritten dataset with the same embedding and reconstruction network. We can observe that (1) trainable fusion layers outperform the hand-designed rules, and our MetaViewer yields the best performance; (2) the MSE scores listed in the last column indicate that the quality of the unified representation cannot be measured and guaranteed only with the reconstruction constraint, due to the view-private redundant information mixed in view-specific latent features.

To further explore the fusion preferences learned by MetaViewer, we visualize the class activation map for the RGB-D samples in Fig. 4. MetaViewer balances information from all views instead of just the salient one (view 1), ensuring a high-quality unified multi-view representation.

4.4. Ablation Studies

Meta-learner structures. We implement the meta-learner as a channel-level convolution structure in this work. Albeit simple, this layer can be considered as a universal approximator for almost any continuous function [40], and thus can fit a wide range of conventional fusion functions. To investigate the effect of network depth, width, and convolution kernel size on the performance of the representation, we alternately fix the 32 kernels and the 1×3 kernel size and show the classification results on Handwritten data in Fig. 5. It is clear that (1) the meta-learner works well with just a shallow structure, as shown in Fig. 5 (a), instead of gradually overfitting to the training data as the network deepens or widens, and (2) our MetaViewer is stable and insensitive to the hyper-parameters within reasonable ranges.

Meta-split ratios. Fig. 6 (a) shows the impact of the meta-split mentioned in Sec. 3.2 on the classification performance, where the proportion of support set is set from 0.1 to 0.9 in steps of 0.1, and the rest is query set. In addition to the single view, we also compare the *sum* and *concat.* fusion as baselines. MetaViewer consistently surpasses all baselines over the experimental proportion. In addition, fusion baselines are more dependent on the better-performing view at lower proportions, instead becoming unstable as the available query sample decreases.

Inner-level update steps. Another hyper-parameter is the number of iteration steps in inner-level optimization. More iterations mean a larger gap from the learned meta representation to the specific view space, i.e., coarser modeling of view-private information. Fig. 6 (b) shows the classification results with various steps, where n steps mean that the inner-level optimization is updated n times throughout the training. MetaViewer achieves the best results when using 1 steps, and remains stable within 15 steps.

5. Conclusion

This work introduced a novel meta-learning perspective for multi-view learning, and proposed a meta-learner, namely MetaViewer, to derive a high-quality unified representation for downstream tasks. In contrast to the prevailing *specific-to-uniform* pipeline, MetaViewer observes the reconstruction process from unified representation to specific views and essentially learns an optimal data-driven fusion that separates and filters out meaningless view-private information. The proposed framework is compatible with most excellent objective functions and pre-text tasks to cope with complex real-world scenarios. Extensive experimental results on clustering and classification tasks demonstrate the performance of meta-learned unified representation. In addition, we believe that our bi-level training pipeline is also promising for handling incomplete views, and leave it to future work.

Acknowledgements

This research was supported in part by the Natural Science Foundation of China (No. 62106129, 62176139, and 62177031), the Natural Science Foundation of Shandong Province (No. ZR2021QF053, ZR2021ZD15), and the China Postdoctoral Science Foundation (No. 2021TQ0195, 2021M701984). The author acknowledges Prof. Xiushan Nie of Shandong Jianzhu University (China) for the financial support and valuable discussions.

References

[1] Sara Atito Ali Ahmed, Muhammad Awais, and Josef Kittler. Sit: Self-supervised vision transformer. *CoRR*, abs/2104.03602, 2021. 4

[2] Galen Andrew, Raman Arora, Jeff A. Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *ICML*, volume 28, pages 1247–1255, 2013. 2, 6, 7

[3] Philip Bachman, R. Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *NeurIPS*, pages 15509–15519, 2019. 2

[4] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, pages 177–186, 2010. 4

[5] Abhra Chaudhuri, Massimiliano Mancini, Yanbei Chen, Zeynep Akata, and Anjan Dutta. Cross-modal fusion distillation for fine-grained sketch-based image retrieval. In *BMVC*, page 499, 2022. 2

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, volume 119, pages 1597–1607, 2020. 5

[7] Yanhua Cheng, Xin Zhao, Rui Cai, Zhiwei Li, Kaiqi Huang, and Yong Rui. Semi-supervised multimodal deep learning for RGB-D object recognition. In *IJCAI*, pages 3345–3351, 2016. 2

[8] Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *ICLR*, 2020. 1, 6, 7

[9] Li Fei-Fei, Robert Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Comput. Vis. Image Underst.*, 106(1):59–70, 2007. 6

[10] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, pages 1933–1941, 2016. 2

[11] Fangxiang Feng, Xiaojie Wang, and Ruifan Li. Cross-modal retrieval with correspondence autoencoder. In *ACM MM*, pages 7–16, 2014. 2

[12] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. GVCNN: group-view convolutional neural networks for 3d shape recognition. In *CVPR*, pages 264–272, 2018. 2

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70, pages 1126–1135, 2017. 3

[14] Kuiliang Gao, Bing Liu, Xuchu Yu, and Anzhu Yu. Unsupervised meta learning with multiview constraints for hyperspectral image small sample set classification. *IEEE TIP*, 31:3449–3462, 2022. 3

[15] Quanxue Gao, Wei Xia, Zhizhen Wan, De-Yan Xie, and Pu Zhang. Tensor-svd based graph learning for multi-view subspace clustering. In *AAAI*, pages 3930–3937, 2020. 6, 7

[16] Yu Geng, Zongbo Han, Changqing Zhang, and Qinghua Hu. Uncertainty-aware multi-view representation learning. In *AAAI*, pages 7545–7553, 2021. 1

[17] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, volume 15, pages 315–323, 2011. 4

[18] Kaveh Hassani and Amir Hosein Khas Ahmadi. Contrastive multi-view representation learning on graphs. In *ICML*, volume 119, pages 4116–4126, 2020. 2, 5

[19] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. Meta-learning in neural networks: A survey. *IEEE TPAMI*, 44(9):5149–5169, 2022. 2

[20] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Sharable and individual multi-view metric learning. *IEEE TPAMI*, 40(9):2281–2288, 2018. 2

[21] Mike Huisman, Jan N. van Rijn, and Aske Plaat. A survey of deep meta-learning. *Artif. Intell. Rev.*, 54(6):4483–4541, 2021. 2

[22] Xiaodong Jia, Xiao-Yuan Jing, Xiaoke Zhu, Songcan Chen, Bo Du, Ziyun Cai, Zhenyu He, and Dong Yue. Semi-supervised multi-view deep discriminant representation learning. *IEEE TPAMI*, 43(7):2496–2509, 2021. 2

[23] Yuheng Jia, Hui Liu, Junhui Hou, Sam Kwong, and Qingfu Zhang. Multi-view spectral clustering tailored tensor low-rank representation. *IEEE TCSVT*, 31(12):4784–4797, 2021. 6, 7

[24] Xiao-Yuan Jing, Fei Wu, Xiwei Dong, Shiguang Shan, and Songcan Chen. Semi-supervised multi-view correlation feature learning with application to webpage classification. In *AAAI*, pages 1374–1381, 2017. 2

[25] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *IEEE TPAMI*, 39(4):664–676, 2017. 2

[26] Yingming Li, Ming Yang, and Zhongfei Zhang. A survey of multi-view representation learning. *IEEE TKDE*, 31(10):1863–1883, 2019. 2, 7

[27] Yijie Lin, Yuanbiao Gou, Xiaotian Liu, Jinfeng Bai, Jiancheng Lv, and Xi Peng. Dual contrastive prediction for incomplete multi-view representation learning. *IEEE TPAMI*, 2022. 1, 2, 5, 6, 7

[28] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *ICCV*, pages 3295–3304, 2019. 3

[29] Xiankai Lu, Wenguan Wang, Chao Ma, Jianbing Shen, Ling Shao, and Fatih Porikli. See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *CVPR*, pages 3623–3632, 2019. 2

[30] Shirui Luo, Changqing Zhang, Wei Zhang, and Xiaochun Cao. Consistent and specific multi-view subspace clustering. In *AAAI*, pages 3730–3737, 2018. 2

[31] Yao Ma, Shilin Zhao, Weixiao Wang, Yaoman Li, and Irwin King. Multimodality in meta-learning: A comprehensive survey. *KBS*, 250:108976, 2022. 3

[32] Antoine Miech, Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Andrew Zisserman. Thinking fast and slow: Efficient text-to-visual retrieval with transformers. In *CVPR*, pages 9826–9836, 2021. 2

[33] Feiping Nie, Guohao Cai, Jing Li, and Xuelong Li. Auto-weighted multi-view learning for image clustering and semi-supervised classification. *IEEE TIP*, 27(3):1501–1511, 2018. 2

[34] Xiushan Nie, Weizhen Jing, Chaoran Cui, Chen Jason Zhang, Lei Zhu, and Yilong Yin. Joint multi-view hashing for large-scale near-duplicate video retrieval. *IEEE TKDE*, 32(10):1951–1965, 2020. 2

[35] Xiushan Nie, Xingbo Liu, Jie Guo, Letian Wang, and Yilong Yin. Supervised discrete multiple-length hashing for image retrieval. *IEEE TBD*, 9(1):312–327, 2023. 1

[36] Zheyun Qin, Xiankai Lu, Xiushan Nie, Dongfang Liu, Yilong Yin, and Wenguan Wang. Coarse-to-fine video instance segmentation with factorized conditional appearance flows. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1–17, 2023. 2

[37] Mathieu Salzmann, Carl Henrik Ek, Raquel Urtasun, and Trevor Darrell. Factorized orthogonal latent spaces. In *AISTATS*, volume 9, pages 701–708, 2010. 2

[38] Yang Shi, Xiushan Nie, Xingbo Liu, Lu Yang, and Yilong Yin. Zero-shot hashing via asymmetric ratio similarity matrix. *IEEE TKDE*, 2022. 2

[39] Yang Shi, Xiushan Nie, Xingbo Liu, Li Zou, and Yilong Yin. Supervised adaptive similarity matrix hashing. *IEEE TIP*, 31:2755–2766, 2022. 2

[40] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, pages 1917–1928, 2019. 3, 8

[41] Haoliang Sun, Chenhui Guo, Qi Wei, Zhongyi Han, and Yilong Yin. Learning to rectify for robust learning with noisy labels. *PR*, 124:108467, 2022. 3

[42] Kai Sun, Jiangshe Zhang, Junmin Liu, Ruixuan Yu, and Zengjie Song. DRCNN: dynamic routing convolutional neural network for multi-view 3d object recognition. *IEEE TIP*, 30:868–877, 2021. 2

[43] Shiliang Sun, Wenbo Dong, and Qiuyang Liu. Multi-view representation learning with deep gaussian processes. *IEEE TPAMI*, 43(12):4453–4468, 2021. 1

[44] Zhongkai Sun, Prathusha Kameswara Sarma, William A. Sethares, and Yingyu Liang. Learning relationships between text, audio, and video via deep canonical correlation for multimodal language analysis. In *AAAI*, pages 8992–8999, 2020. 2

[45] Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J. Lim. Multimodal model-agnostic meta-learning via task-aware modulation. In *NeurIPS*, pages 1–12, 2019. 3

[46] Shiye Wang, Changsheng Li, Yanming Li, Ye Yuan, and Guoren Wang. Self-supervised information bottleneck for deep multi-view subspace clustering. *CoRR*, abs/2204.12496, 2022. 6

[47] Weiran Wang, Raman Arora, Karen Livescu, and Jeff A. Bilmes. On deep multi-view representation learning. In *ICML*, volume 37, pages 1083–1092, 2015. 2, 5, 6, 7

[48] Qi Wei, Lei Feng, Haoliang Sun, Ren Wang, Chenhui Guo, and Yilong Yin. Fine-grained classification with noisy labels. *arXiv preprint arXiv:2303.02404*, 2023. 3

[49] Qi Wei, Haoliang Sun, Xiankai Lu, and Yilong Yin. Self-filtering: A noise-aware sample selection for label noise with confidence penalization. In *ECCV*, pages 516–532, 2022. 2

[50] Jie Xu, Yazhou Ren, Huayi Tang, Zhimeng Yang, Lili Pan, Yang Yang, and Xiaorong Pu. Self-supervised discriminative feature learning for multi-view clustering. *CoRR*, abs/2103.15069, 2021. 6

[51] Jie Xu, Huayi Tang, Yazhou Ren, Liang Peng, Xiaofeng Zhu, and Lifang He. Multi-level feature learning for contrastive multi-view clustering. In *CVPR*, pages 16030–16039, 2022. 1, 2, 5, 6, 7

[52] Fei Xue, Xin Wu, Shaojun Cai, and Junqiu Wang. Learning multi-view camera relocalization with graph neural networks. In *CVPR*, pages 11372–11381, 2020. 2

[53] Xiaowei Xue, Feiping Nie, Sen Wang, Xiaojun Chang, Bela Stantic, and Min Yao. Multi-view correlated feature learning by uncovering shared component. In *AAAI*, pages 2810–2816, 2017. 2

[54] Xiaoqiang Yan, Shizhe Hu, Yiqiao Mao, Yangdong Ye, and Hui Yu. Deep multi-view learning methods: A review. *Neurocomputing*, 448:106–129, 2021. 2

[55] En Yu, Zhuoling Li, and Shoudong Han. Towards discriminative representation: Multi-view trajectory contrastive learning for online multi-object tracking. In *CVPR*, pages 8824–8833, 2022. 2, 4, 5

[56] Xiao Yu, Hui Liu, Yuxiu Lin, Yan Wu, and Caiming Zhang. Auto-weighted sample-level fusion with anchors for incomplete multi-view clustering. *PR*, 130:108772, 2022. 2

[57] Yun-Hao Yuan, Jin Li, Yun Li, Jipeng Qiang, Yi Zhu, Xiaobo Shen, and Jianping Gou. Learning canonical f-correlation projection for compact multiview representation. In *CVPR*, pages 19238–19247, 2022. 2, 4

[58] Changqing Zhang, Huazhu Fu, Qinghua Hu, Xiaochun Cao, Yuan Xie, Dacheng Tao, and Dong Xu. Generalized latent multi-view subspace clustering. *IEEE TPAMI*, 42(1):86–99, 2020. 1

[59] Changqing Zhang, Zongbo Han, Yajie Cui, Huazhu Fu, Joey Tianyi Zhou, and Qinghua Hu. Cpm-nets: Cross partial multi-view networks. In *NeurIPS*, pages 557–567, 2019. 6

[60] Changqing Zhang, Yeqing Liu, and Huazhu Fu. Ae2-nets: Autoencoder in autoencoder networks. In *CVPR*, pages 2577–2585, 2019. 2, 5, 6

[61] Liang Zhao, Tao Yang, Jie Zhang, Zhikui Chen, Yi Yang, and Z. Jane Wang. Co-learning non-negative correlated and uncorrelated features for multi-view data. *IEEE TNNLS*, 32(4):1486–1496, 2021. 2

[62] Wenqing Zheng, Tianlong Chen, Ting-Kuei Hu, and Zhangyang Wang. Symbolic learning to optimize: Towards interpretability and scalability. In *ICLR*, 2022. 3

[63] Pengfei Zhu, Binyuan Hui, Changqing Zhang, Dawei Du, Longyin Wen, and Qinghua Hu. Multi-view deep subspace clustering networks. *CoRR*, abs/1908.01978, 2019. 6