Multiple View Clustering Using a Weighted Combination of Exemplar-Based Mixture Models

Grigorios F. Tzortzis and Aristidis C. Likas, *Senior Member, IEEE*

***Abstract*— Multiview clustering partitions a dataset into groups by simultaneously considering multiple representations (views) for the same instances. Hence, the information available in all views is exploited and this may substantially improve the clustering result obtained by using a single representation. Usually, in multiview algorithms all views are considered equally important, something that may lead to bad cluster assignments if a view is of poor quality. To deal with this problem, we propose a method that is built upon exemplar-based mixture models, called convex mixture models (CMMs). More specifically, we present a multiview clustering algorithm, based on training a weighted multiview CMM, that associates a weight with each view and learns these weights automatically. Our approach is computationally efficient and easy to implement, involving simple iterative computations. Experiments with several datasets confirm the advantages of assigning weights to the views and the superiority of our framework over single-view and unweighted multiview CMMs, as well as over another multiview algorithm which is based on kernel canonical correlation analysis.**

***Index Terms*— Clustering, mixture models, multiview learning.**

I. Introduction

THE MOST common approach for the machine learning and data mining settings is to assume that data are represented in a single vector or graph space. However, in many real-life problems multiview data arise naturally. Multiview data are instances that have multiple representations (views) from different feature spaces. Usually, these multiple views are from different vector spaces or different graph spaces or a combination of vector and graph spaces. The most typical example is web pages. Web pages can be represented with a term vector whose elements correspond to the occurrence of certain words in the web page text, a hyperlink graph that shows to which other web pages each web page points to and also a term vector for the words in the anchor text. Another example is scientific articles, which can be represented with a term vector corresponding to the words appearing in the abstract and introduction and also a citation graph.

The natural and frequent occurrence of multiview data has raised interest in the so-called *multiview learning*. The main challenge of multiview learning is to develop algorithms that use multiple views simultaneously, given the diversity of the views. Most studies on this topic address the semisupervised classification problem, and multiview classification algorithms have often proven to utilize unlabeled data effectively and substantially improve classification accuracy (e.g., [1]–[3]).

This paper focuses on multiview unsupervised learning and particularly in *multiview clustering*. Multiview clustering explores and exploits multiple representations simultaneously in order to produce a more accurate and robust partitioning of the data than single-view clustering. The intuition behind this approach is that the different representations are more informative regarding the correct partitioning of the dataset than a single view. Therefore, by taking advantage of all the available views, we expect to produce a better splitting of the data. The available literature for this topic is growing fast (e.g., [4]–[11]), with encouraging results. Borrowing the terminology of [9], there exist two approaches in multiview clustering: *centralized* and *distributed*. Centralized algorithms simultaneously use all available views to cluster the dataset, while distributed algorithms first cluster each view independently from the others, using an appropriate single-view algorithm, and then combine the individual clusterings to produce a final partitioning.

Most multiview algorithms rely equally on every view in order to compute a clustering, but the useful information conveyed by the available views can vary significantly. For example, one view may contain noise, or outliers, or be irrelevant to the task in hand. Including such a view in the partitioning process may result in performance degradation. Identifying and removing such views beforehand is not easy though. For this reason, we present a multiview centralized clustering method that *assigns different weights to the views and learns those weights automatically*. The weights reflect the quality of each view and therefore affect its contribution to the final clustering solution accordingly. Specifically, we extend our previous framework [10], namely multiview convex mixture models (CMMs), to accommodate weights for the views.

Multiview CMMs generalize a special case of mixture models, called CMMs[1] (a.k.a. exemplar-based mixture models) [12], to data with multiple representations. They locate exemplars in the dataset (i.e., instances that serve as the representatives of the clusters) through a convex optimization by equally considering all available views. Also, the different statistical properties of the views are taken into account, and good results have been achieved in [10]. The proposed

Manuscript received December 2, 2009; revised September 21, 2010; accepted September 22, 2010. Date of publication October 7, 2010; date of current version November 30, 2010.

The authors are with the Department of Computer Science, University of Ioannina, Ioannina 45110, Greece (e-mail: gtzortzi@cs.uoi.gr; arly@cs.uoi.gr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2010.2081999

[1]We shall refer to CMMs as single-view CMMs whenever it is necessary to make the distinction to their multiview counterparts explicit.

weighted multiview CMM is a weighted combination of CMMs (one for each view) that identifies exemplars in the dataset. It incorporates most of the advantages of multiview CMMs, plus the ability to spot irrelevant views through the weights. As we shall see, this model has also a probabilistic explanation.

The experiments with our algorithm on noisy artificial datasets and two real datasets demonstrate in most cases a considerable improvement on the clustering performance when compared to: 1) the single-view CMM [12] applied on the individual views; 2) the single-view CMM that uses the concatenation of the views; 3) the multiview CMM [10]; and 4) the multiview clustering method of [5] which first combines the views through kernel canonical correlation analysis (Kernel CCA) projection and then clusters the derived projections. These results confirm the effectiveness of our model as well as its ability to correctly measure the quality of the views and adjust the weights so as to get good clustering solutions not affected by the presence of noisy or noninformative views.

The rest of this paper is organized as follows. The next section reviews related work, while Section III provides an analysis on single-view and multiview CMMs. The proposed algorithm follows in Section IV. The experimental evaluation on artificial and real data is discussed in Section V. Finally, Section VI concludes this paper.

II. Related Work

Multiview learning in the semisupervised setting has been introduced by Yarowsky [13] and Blum and Mitchell [1]. In [13], a two-view word sense disambiguation algorithm was described, which uses two classifiers that bootstrap each other. Blum and Mitchel [1] introduced the co-training algorithm to train a classifier from two representations. Their idea is to train two learners on distinct views of the labeled data and iteratively allow each learner to label the unlabeled instances that predicts with the highest confidence. Assuming the learners are independent, the newly labeled instances may help the other learner to improve its model. The co-training method was modified in [14], so that an objective function that measures the degree of agreement between the two views is optimized. Zhou and Li [15] proposed co-training for regression. PAC bounds on the error of co-training were given in [16] and [17], showing that the disagreement of two learners in two independent views upper-bounds the error rate. The co-EM algorithm [18] is based on the co-training idea and is a variant of EM for two-view semisupervised learning, for which many extensions have been proposed [2], [3], [19].

Most of the existing work in multiview clustering follows the centralized approach and extends well-known clustering algorithms to the multiview setting. Bickel and Scheffer [4] developed a two-view EM and a two-view k-means algorithm under the assumption that the two views are independent. They also studied the problem of mixture model estimation with more than two views and showed that co-EM is a special case of their formulation [20]. De Sa [7] proposed a two-view spectral clustering algorithm that creates a bipartite graph of the views and is based on the "minimizing-disagreement" idea [21], [22]. This method also assumes that the views are independent. An algorithm that generalizes the single-view normalized cut to the multiview case and can handle both directed and undirected graphs was introduced by Zhou and Burges [11]. Their idea can be explained as a vertex-wise mixture of Markov chains associated with different graphs and is applicable to more than two views. Blaschko and Lampert [5] projected the data onto the top directions obtained by kernel CCA across the views and applied k-means to cluster the projections. Finally, in [6] each of the two views was assumed to be generated by a mixture of distributions and CCA was employed to project the data to the subspace spanned by the distributions' means. Afterwards, a standard clustering algorithm was used in this space to split the data. Theoretical results were provided that guarantee that the method can recover the correct clusters with high probability if an independence assumption among the two views and a rank assumption on the CCA matrix hold, as well as a separation condition.

Following the distributed approach, Long *et al*. [9] proposed a general model for multiview unsupervised learning. According to their model, the final partitioning of the data is derived by minimizing an appropriate objective function that measures how close the final clustering, based on all views, is to the clustering of each single view with the help of a mapping function. In [8], a matrix factorization approach was adopted to reconcile the groups arising from the individual views. Specifically, a matrix that contains the partitioning of every individual view is created and then decomposed to two matrices, the one showing the contribution of those partitionings to the final "multiview" clusters, called meta-clusters, and the other the assignment of instances to the meta-clusters. Both [8] and [9] can handle any number of views and representations from both vector and graph spaces.

A machine learning problem that is closely related to multiview clustering, although seemingly different, is *unsupervised multiple kernel learning*, where multiple kernels are available for the same instances (the dataset has only one view) and an appropriate combination of those kernels is learned within a clustering framework. If the kernel matrix is given for each view, we can apply unsupervised multiple kernel learning techniques to partition multiview data. Examples of such algorithms are the ones based on the ideas of maximum margin clustering [23]–[25] and local learning clustering [26].

III. Exemplar-Based Mixture Models

This section briefly describes exemplar-based mixture models [12], also known as *CMMs*, since they consist a key part of our new algorithm and also our previous multiview clustering work [10] that is built upon CMMs and is extended by the current framework.

A. CMMs

CMMs [12] are simplified mixture models that result in soft assignments of data points to clusters and in the extraction of representative exemplars from the dataset. When training these models, which is done by maximizing the log-likelihood,

all instances compete to become the "center" of the clusters (i.e., cluster representatives-exemplars), since *the number of the CMM components is equal to the number of data points and each component distribution is centered at a distinct dataset point*. In the end, the instances corresponding to the components that have received during training the highest priors are selected as exemplars and the remaining instances are assigned to the "closest" exemplar. Note that the *priors of the components are the only adjustable parameters of a CMM*.

In detail, given a dataset $\mathfrak{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \Re^d$, the CMM distribution is $Q(\mathbf{x}) = \sum_{j=1}^{N} q_j f_j(\mathbf{x})$, $\mathbf{x} \in \Re^d$, where $q_j \geq 0$ denotes the prior probability of the jth component, satisfying the constraint $\sum_{j=1}^{N} q_j = 1$, and $f_j(\mathbf{x})$ is an exponential family distribution, with its expectation parameter equal to the jth data point. Note that the same exponential family is used for all components. Taking into account the bijection between regular exponential families and Bregman divergences [27], we write $f_j(\mathbf{x}) = C_\varphi(\mathbf{x}) \exp(-\beta d_\varphi(\mathbf{x}, \mathbf{x}_j))$, with d_φ denoting the Bregman divergence corresponding to the components' distributions, $C_\varphi(\mathbf{x})$ being independent of $\mathbf{x}_j$, and β being a constant controlling the sharpness of the components [12].

A clustering is produced by maximizing the (exemplar-based) dataset log-likelihood $L(\mathfrak{X}; \{q_j\}_{j=1}^N)$, defined in (1), over $\{q_j\}_{j=1}^N$, s.t. $\sum_{j=1}^N q_j = 1$

$$L\left(\mathfrak{X}; \{q_j\}_{j=1}^N\right) = \frac{1}{N}\sum_{i=1}^{N} \log\left[\sum_{j=1}^{N} q_j f_j(\mathbf{x}_i)\right] = \frac{1}{N}\sum_{i=1}^{N} \log\left[\sum_{j=1}^{N} q_j e^{-\beta d_\varphi(\mathbf{x}_i, \mathbf{x}_j)}\right] + \text{const.} \tag{1}$$

If we define $\hat{P}(\mathbf{x}) = 1/N$, $\mathbf{x} \in \mathfrak{X}$ to be the empirical dataset distribution, we can equivalently formulate the above likelihood maximization problem in terms of the Kullback–Leibler (KL) divergence among $\hat{P}(\mathbf{x})$ and $Q(\mathbf{x})$, since their KL distance is

$$D(\hat{P} \| Q) = -\sum_{i=1}^{N} \hat{P}(\mathbf{x}_i) \log Q(\mathbf{x}_i) - \mathbb{H}(\hat{P}) = -L\left(\mathfrak{X}; \{q_j\}_{j=1}^N\right) + \text{const.} \tag{2}$$

where $\mathbb{H}(\hat{P})$ is the entropy of the empirical distribution $\hat{P}(\mathbf{x})$ which does not depend on the parameters q_j of the CMM. Now, instead of maximizing (1), we can minimize (2). This minimization problem is *convex* and can be solved with an iterative algorithm, whose updates for the components' prior probabilities are given by

$$q_j^{(t+1)} = q_j^{(t)} \sum_{i=1}^{N} \frac{\hat{P}(\mathbf{x}_i) f_j(\mathbf{x}_i)}{\sum_{j'=1}^{N} q_{j'}^{(t)} f_{j'}(\mathbf{x}_i)} \tag{3}$$

and the algorithm is *guaranteed to converge to the global minimum* as long as $q_j^{(0)} > 0, \forall j$ [12]. Note, that the prior probability q_j associated with data point $\mathbf{x}_j$ is a measure of *how likely this point is to be an exemplar*.

The ability of always being able to locate the global optimum makes this model attractive, as it avoids the initialization and local optima problems of standard mixture models, which require multiple executions of the EM algorithm [28]. Moreover, in [12] a Gaussian CMM was found to be experimentally superior to a fully parameterized Gaussian mixture model, despite its smaller flexibility as q_j are the only parameters. Another important feature is that only the pairwise data distances $d_\varphi(\mathbf{x}_i, \mathbf{x}_j)$ take part in the calculation of the priors, as $C_\varphi(\mathbf{x}_i)$ cancels out, thus the values of the data points are not required if we are given the distances.

Splitting the dataset into M disjoint clusters is done by requiring the instances with the M highest q_j values to serve as exemplars and then assigning the remaining instances to the exemplar with the highest posterior probability.

Finally, clustering with a CMM requires the selection of an appropriate value for the constant β ($0 < \beta < \infty$). It is possible to identify a reasonable range of β values by determining a reference value β_0. In [12], the following empirical rule (4) has been adopted, achieving good results in the experiments

$$\beta_0 = N^2 \log N / \sum_{i,j=1}^{N} d_\varphi(\mathbf{x}_i, \mathbf{x}_j). \tag{4}$$

B. Multiview CMMs

Multiview CMMs [10] assume that the representations of the instances in each view are generated by a CMM and aim at locating good exemplars by considering all available views. More specifically, if we are given a dataset with N instances and V views, $\mathfrak{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i$ contains the representations of the ith instance across the views, i.e., $\boldsymbol{x}_i = \{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^V\}$, $\mathbf{x}_i^v \in \Re^{d^v}$, the mixture distribution of each view is

$$Q^v(\mathbf{x}^v) = \sum_{j=1}^{N} q_j f_j^v(\mathbf{x}^v) = C_{\varphi_v}(\mathbf{x}^v) \sum_{j=1}^{N} q_j e^{-\beta^v d_{\varphi_v}(\mathbf{x}^v, \mathbf{x}_j^v)}, \qquad \mathbf{x}^v \in \Re^{d^v}. \tag{5}$$

Since the statistical properties of individual views may differ substantially, different views are allowed to have component distributions $f_j^v(\mathbf{x}^v)$ from different exponential families. On the other hand, as a clustering based on all views is pursued, all $Q^v(\mathbf{x}^v)$ share the same priors in order to interact.

Moreover, an empirical dataset distribution $\hat{P}^v(\mathbf{x}^v) = 1/N, \mathbf{x}^v \in \{\mathbf{x}_1^v, \mathbf{x}_2^v, \ldots, \mathbf{x}_N^v\}$, is associated with every view and the multiview algorithm minimizes the sum of KL divergences

(6) between $\hat{P}^v(\mathbf{x}^v)$ and $Q^v(\mathbf{x}^v)$ across all views

$$\min_{\substack{q_1,\ldots,q_N \\ \text{s.t. } \sum_{j=1}^N q_j=1}} \left\{ \sum_{v=1}^{V} D(\hat{P}^v \| Q^v) \right\} = \min_{\substack{q_1,\ldots,q_N \\ \text{s.t. } \sum_{j=1}^N q_j=1}} \left\{ -\sum_{v=1}^{V}\sum_{i=1}^{N} \hat{P}^v(\mathbf{x}_i^v) \log Q^v(\mathbf{x}_i^v) - \sum_{v=1}^{V} \mathbb{H}(\hat{P}^v) \right\}. \tag{6}$$

It is quite straightforward to see that the optimized objective is *convex*, hence the global minimum is always found, and the rule for updating the priors (7) is a generalization of the single-view case. Once again, for the updates only the pairwise distances in each view are required, as $C_{\varphi_v}(\mathbf{x}_i^v)$ cancels out

$$q_j^{(t+1)} = \frac{q_j^{(t)}}{V} \sum_{v=1}^{V}\sum_{i=1}^{N} \frac{\hat{P}^v(\mathbf{x}_i^v) f_j^v(\mathbf{x}_i^v)}{\sum_{j'=1}^N q_{j'}^{(t)} f_{j'}^v(\mathbf{x}_i^v)}. \tag{7}$$

The prior q_j associated with the jth instance is a measure of *how likely this instance is to be an exemplar*, taking into account all views.

Splitting the dataset into M disjoint clusters is done by identifying the instances with the M highest q_j values (these are the exemplars) and then assigning the remaining instances to the exemplar with the highest posterior probability over all views. Appropriate β^v values are found in the range of an empirically defined β_0^v value (8)

$$\beta_0^v = N^2 \log N / \sum_{i,j=1}^{N} d_{\varphi_v}\left(\mathbf{x}_i^v, \mathbf{x}_j^v\right). \tag{8}$$

IV. Weighted Multiview CMMs

Motivated by the promising results of CMMs in [10], this paper proposes an alternative centralized scheme to multiview clustering, where *weights are assigned to the views and adjusted through training*.

A. Model Description

From the objective (6) in Section III-B, it can be observed that all views contribute equally to the sum, regardless of how "good" each view is for the problem in hand. Our aim is to locate exemplars in the dataset by *allowing the views to participate with different weights* to the objective function, measuring how "informative" the corresponding view is, and by *learning those weights automatically*, i.e., as part of the learning process. Such an approach generalizes the previous one and could be helpful in cases where a view is irrelevant to the clustering task, or contains noise.

To accomplish our objective, we introduce a weighted combination of exemplar-based models. For the vth view, a CMM $Q^v(\mathbf{x}^v)$, of the same form as in (5), is defined and a positive weight π^v is associated with it. The views are combined by summing the corresponding weighted CMMs.

In more detail, suppose we are given a dataset with N instances and V views, $\mathfrak{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i = \left\{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^V\right\}$, $\mathbf{x}_i^v \in \Re^{d^v}$. Our model, which we will refer to as *weighted multiview CMM*, is given by

$$F\left(\boldsymbol{x} = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^V\right\}\right) = \sum_{v=1}^{V} \pi^v Q^v(\mathbf{x}^v) = \sum_{v=1}^{V} \pi^v \sum_{j=1}^{N} q_j f_j^v(\mathbf{x}^v), \qquad \mathbf{x}^v \in \Re^{d^v} \tag{9}$$

where

$$f_j^v(\mathbf{x}^v) = C_{\varphi_v}(\mathbf{x}^v) e^{-\beta^v d_{\varphi_v}(\mathbf{x}^v, \mathbf{x}_j^v)},$$
$$\pi^v \geq 0, \qquad \sum_{v=1}^{V} \pi^v = 1, \qquad q_j \geq 0, \qquad \sum_{j=1}^{N} q_j = 1.$$

Note the imposed constraints on the weights π^v. Due to these restrictions, $F(\boldsymbol{x})$ has a probabilistic interpretation. Specifically, it is a mixture model whose number of components is equal to the number of the views and each component is a CMM $Q^v(\mathbf{x}^v)$, corresponding to the vth view. Hence, *the weights can also be seen as the prior probabilities of the views* under the mixture model.

The above formulation has some very important characteristics. To capture the diversity among the views, they are allowed to have distributions $f_j^v(\mathbf{x}^v)$ coming from different exponential families, i.e., have different β^v values and Bregman divergences d_{φ_v}. For example, a Gaussian CMM can be used for one view and a Bernoulli CMM for another. Since a CMM is used for each view, all instances will be considered as possible cluster centroids (i.e., exemplars) during training. Moreover, the priors q_j are the same across all views, to allow the extraction of representative exemplars based on every view. Therefore, an instance whose corresponding prior q_j has a high value will more or less be a good exemplar for all views. Finally, a low π^v value indicates that view v conveys little information regarding the partitioning of the dataset.

B. Model Training and Multiview Clustering

Since $F(\boldsymbol{x})$ can be viewed as a mixture model, to partition the dataset $\mathfrak{X}$ we must maximize the log-likelihood (10) w.r.t. the parameters $\{\pi^v\}_{v=1}^V$, $\{q_j\}_{j=1}^N$, s.t. the constraints $\sum_{v=1}^V \pi^v = 1$, $\sum_{j=1}^N q_j = 1$. It must be stressed that, in contrast to multiview CMMs [10], this optimization task is *not convex* due to the introduction of the weights π^v. However, we hope to compensate for the lost convexity by the ability to estimate different weights for the views

$$L\left(\mathfrak{X}; \{\pi^v\}_{v=1}^V, \{q_j\}_{j=1}^N\right) = \sum_{i=1}^{N} \log \sum_{v=1}^{V} \pi^v Q^v(\mathbf{x}_i^v) = \sum_{i=1}^{N} \log\left(\sum_{v=1}^{V} \pi^v \sum_{j=1}^{N} q_j f_j^v(\mathbf{x}_i^v) \right). \tag{10}$$

Local maxima of the log-likelihood can be found by applying the EM algorithm [28]. This algorithm uses an

initial guess for the parameters and iteratively adjusts them, such that the likelihood always increases until a local optimum is reached. Our model has only prior probabilities that can be adjusted, and hence initializing them uniformly and *avoiding multiple restarts for EM* is a natural choice (this approach is followed in the experiments), i.e., $\pi^{v(0)} = 1/V, q_j^{(0)} = 1/N$. Of course, if prior knowledge for the quality of the views exists, this can be directly incorporated into the optimization by initializing π^v accordingly. To briefly illustrate the steps of EM, define $\{\mathfrak{X}, \mathcal{Z}\}$ to be the complete dataset, where $\mathcal{Z} = \{z_1, z_2, \ldots, z_N\}$ contains the latent variables indicating the mixture component responsible for generating each instance, i.e., $z_i \in \{1, 2, \ldots, V\}$. The analytical derivation of the EM equations can be found in the Appendix.

E-step: In practice, we are not given the complete dataset but only the observations $\mathfrak{X}$. Our state of knowledge for the latent variables is described through the posterior probabilities $P(z_i = v|\boldsymbol{x}_i)$ (11), which at iteration t are calculated $\forall i \in \{1, 2, \ldots, N\}, \forall v \in \{1, 2, \ldots, V\}$ as

$$P^{(t)}(z_i = v|\boldsymbol{x}_i) = \frac{\pi^{v(t)} Q^{v(t)}(\mathbf{x}_i^v)}{\sum_{v=1}^{V} \pi^{v(t)} Q^{v(t)}(\mathbf{x}_i^v)}. \tag{11}$$

M-step: The posterior probabilities of the E-step are useful in estimating new values for the parameters during the M-step. By setting to zero the derivative of the constrained complete dataset log-likelihood expectation (23), under the posterior probabilities distribution, w.r.t. $\{\pi^v\}_{v=1}^V$, $\{q_j\}_{j=1}^N$ and a little manipulation, we get the updates for the parameters (12) and (13)

$$\pi^{v(t+1)} = \frac{1}{N} \sum_{i=1}^{N} P^{(t)}(z_i = v|\boldsymbol{x}_i) \tag{12}$$

$$q_j^{(t'+1)} = \frac{q_j^{(t')}}{N} \sum_{i=1}^{N} \sum_{v=1}^{V} P^{(t)}(z_i = v|\boldsymbol{x}_i) \frac{f_j^v(\mathbf{x}_i^v)}{\sum_{j'=1}^{N} q_{j'}^{(t')} f_{j'}^v(\mathbf{x}_i^v)}. \tag{13}$$

Some remarks on the optimization process, whose pseudocode is illustrated in Algorithm 1, follow. First, the new estimation $q_j^{(t'+1)}$ depends on the previous value $q_j^{(t')}$; therefore, a nested loop in the M-step of the EM algorithm is required to perform multiple updates on q_j for the same set of posterior probability values in order to get $q_j^{(t+1)}$. This loop finishes when the change on q_j values between two iterations is less than a small value ϵ' (line 21). Second, EM terminates when the likelihood between consecutive pairs of E and M steps changes less than a small value ϵ (line 24). Third, we must explicitly incorporate into the calculation of the posterior probabilities (11) the $C_{\varphi_v}(\mathbf{x}_i^v)$ values (for the q_j estimations (13) the $C_{\varphi_v}(\mathbf{x}_i^v)$ values still cancel out). Hence, the pairwise distances alone do not suffice to compute the updates, and the dataset instances are required in the general case, contrary to the single-view and multiview CMMs, but for certain distributions $f_j^v(\mathbf{x}^v)$ this is not necessary, as demonstrated in the experimental section for the Gaussian distribution. Fourth, the same empirical β_0^v values (8) as in Section III-B can be adopted to guide the search for appropriate β^v values. Finally, it is obvious that the view weights π^v are determined automatically during the M-step.

Algorithm 1 EM for Weighted Multiview CMMs

Input: Multiview dataset $\mathfrak{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i = \left\{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^V\right\}$

Output: Model parameter estimation: $\{\pi^v\}_{v=1}^V$, $\{q_j\}_{j=1}^N$

// Initialize the parameters.
1: Set $\pi^{v(0)} = 1/V, \forall v = 1, \ldots, V$
2: Set $q_j^{(0)} = 1/N, \forall j = 1, \ldots, N$
3: Set $t = 0$
4: **repeat**
5: // E-step.
6: **for** $i = 1$ to N **do**
7: **for** $v = 1$ to V **do**
8: $P^{(t)}(z_i = v|\boldsymbol{x}_i) = \frac{\pi^{v(t)} Q^{v(t)}(\mathbf{x}_i^v)}{\sum_{v=1}^{V} \pi^{v(t)} Q^{v(t)}(\mathbf{x}_i^v)} = \frac{\pi^{v(t)} \sum_{j=1}^{N} q_j^{(t)} f_j^v(\mathbf{x}_i^v)}{\sum_{v=1}^{V} \pi^{v(t)} \sum_{j=1}^{N} q_j^{(t)} f_j^v(\mathbf{x}_i^v)}$
9: **end for**
10: **end for**
11: // M-step.
12: **for all** π^v, $v = 1$ to V **do** // Update the weights π^v
13: $\pi^{v(t+1)} = \frac{1}{N} \sum_{i=1}^{N} P^{(t)}(z_i = v|\boldsymbol{x}_i)$
14: **end for**
15: Set $t' = t$
16: **repeat** // Update the priors q_j
17: **for all** q_j, $j = 1$ to N **do**
18: $q_j^{(t'+1)} = \frac{q_j^{(t')}}{N} \sum_{i=1}^{N} \sum_{v=1}^{V} P^{(t)}(z_i = v|\boldsymbol{x}_i) \frac{f_j^v(\mathbf{x}_i^v)}{\sum_{j'=1}^{N} q_{j'}^{(t')} f_{j'}^v(\mathbf{x}_i^v)}$
19: **end for**
20: $t' = t' + 1$
21: **until** $\sum_{j=1}^{N} \left| q_j^{(t')} - q_j^{(t'-1)} \right| < \epsilon'$
22: $t = t + 1$
23: Set $\left\{q_j^{(t)}\right\}_{j=1}^N = \left\{q_j^{(t')}\right\}_{j=1}^N$
24: **until** $\left| L\left(\mathfrak{X}; \left\{\pi^{v(t)}\right\}_{v=1}^V, \left\{q_j^{(t)}\right\}_{j=1}^N\right) - L\left(\mathfrak{X}; \left\{\pi^{v(t-1)}\right\}_{v=1}^V, \left\{q_j^{(t-1)}\right\}_{j=1}^N\right) \right| < \epsilon$
25: **return** $\{\pi^v\}_{v=1}^V = \left\{\pi^{v(t)}\right\}_{v=1}^V$, $\{q_j\}_{j=1}^N = \left\{q_j^{(t)}\right\}_{j=1}^N$

If we wish to split the dataset $\mathfrak{X}$ into M disjoint clusters $C_1, C_2, \ldots, C_M$, after EM termination, we must select the instances that will act as exemplars. For this purpose, the instances with the M highest q_j values are chosen, denoted by the set $\mathfrak{X}^E = \left\{\boldsymbol{x}_1^E, \boldsymbol{x}_2^E, \ldots, \boldsymbol{x}_M^E\right\} \subset \mathfrak{X}$. The remaining $N - M$ instances are assigned to cluster C_k, associated with the kth exemplar, that has the largest posterior probability $P(C_k|\boldsymbol{x}_i)$ (14) (for the proof, see the Appendix). In (14), we refer to the prior and component distribution of the vth view CMM corresponding to exemplar $\boldsymbol{x}_k^E$, as q_k^E and $f_k^{vE}(\mathbf{x}^v)$ respectively. The cluster assignments are given by (15)

$$P(C_k|\boldsymbol{x}_i) = \frac{q_k^E \sum_{v=1}^{V} \pi^v f_k^{vE}(\mathbf{x}_i^v)}{\sum_{v=1}^{V} \pi^v \sum_{j=1}^{N} q_j f_j^v(\mathbf{x}_i^v)} \tag{14}$$

$$C_k = \left\{ \boldsymbol{x}_k^E \right\} \cup \left\{ \boldsymbol{x}_i \middle| P(C_k|\boldsymbol{x}_i) > P(C_l|\boldsymbol{x}_i), \forall l \neq k, \boldsymbol{x}_i \notin \mathfrak{X}^E \right\}. \tag{15}$$

C. Additional Aspects

If the data points of each view are mapped from input space to a higher dimensional feature space, through a nonlinear transformation ϕ^v, our method can be readily applied to the mapped data $\left\{\phi^1(\mathbf{x}_i^1), \phi^2(\mathbf{x}_i^2), \ldots, \phi^V(\mathbf{x}_i^V)\right\}, i = 1, 2, \ldots, N$ and thus *perform multiview clustering in feature space*. By representing the instances in a new space, a clearer group structure can emerge. It is a very common practice in feature space clustering that the mapping function ϕ^v is not explicitly defined. Usually, a kernel function $K^v(\mathbf{x}_i^v, \mathbf{x}_j^v)$ is used, which is applied to the input space instances and directly provides the inner products in feature space, and a kernel matrix $\mathbf{K}^v \in \Re^{N\times N}$, where $\mathbf{K}_{ij}^v = \phi^v(\mathbf{x}_i^v)^T\phi^v(\mathbf{x}_j^v) = K^v(\mathbf{x}_i^v, \mathbf{x}_j^v)$ is built. Hence, if the required calculations involve only inner products in feature space, there is no need to know ϕ^v. This is the case for our approach when considering Gaussian CMMs in the feature space of each view, as $d_{\varphi_v}(\phi^v(\mathbf{x}_i^v), \phi^v(\mathbf{x}_j^v)) = \|\phi^v(\mathbf{x}_i^v) - \phi^v(\mathbf{x}_j^v)\|^2 = \mathbf{K}_{ii}^v + \mathbf{K}_{jj}^v - 2\mathbf{K}_{ij}^v$. Consequently, the required pairwise distances are expressed in terms of the values of the kernel matrices. This also makes our method directly applicable for unsupervised multiple kernel learning [23], in case we are given multiple kernels for the instances of a single-view dataset and wish to find an appropriate combination of the kernels that clusters the data efficiently, i.e., our algorithm will treat each kernel as being a distinct view, will compute the pairwise distances as shown above, and will obtain a combination of the kernels through the weights π^v.

As for the complexity of the EM for our model, the calculation of the posteriors $P(z_i = v|\boldsymbol{x}_i)$ requires $O(N^2V)$ scalar operations, while the updates on the weights π^v and the priors q_j cost $O(NV)$ and $O(N^2V)$ scalar operations, respectively. Assuming τ EM iterations are performed until convergence and τ' iterations in each nested loop of the M-step when estimating the priors q_j, the overall complexity is $O(N^2V\tau + NV\tau + N^2V\tau\tau') = O(N^2V\tau\tau')$. Finally, if we are not given the pairwise distances $d_{\varphi_v}(\mathbf{x}_i^v, \mathbf{x}_j^v)$ of each view, their computation usually costs an extra $O(N^2Vd)$ scalar operations, where $d = \max\{d^1, d^2, \ldots, d^V\}$.

V. Empirical Evaluation

A. Experimental Setup

The performance of the weighted multiview CMM is studied on both synthetic and real data. The real datasets are a collection of academic web pages and a set of images on Internet pages, where multiple views occur naturally.

Our focus is to compare the proposed method to multiview CMMs [10] and to investigate whether assigning different weights to views is helpful. Moreover, a single-view CMM is applied to each of the individual views of the datasets to examine whether multiple views boost the clustering quality. Note that our new algorithm and multiview CMMs reduce to the single-view CMMs of Section III-A when only one view is present. Since the easiest way to partition data with multiple representations is to concatenate the views (e.g., by appending the vectors) and then apply a single-view algorithm on this concatenation, the single-view CMM is also tested using the concatenated view, in order to explore whether the weighted multiview approach leads to improved performance.

Gaussian CMMs are adopted for all cases and views, i.e., $d_{\varphi_v}(\mathbf{x}_i^v, \mathbf{x}_j^v) = \|\mathbf{x}_i^v - \mathbf{x}_j^v\|^2$. For Gaussian components, $C_{\varphi_v}(\mathbf{x}_i^v) = (\beta^v/2\pi)^{d^v/2}$, hence $C_{\varphi_v}(\mathbf{x}_i^v)$ does not depend on the instance values $\mathbf{x}_i^v$. Therefore, the pairwise distances along with each view dimensionality d^v suffice to calculate the updates for our method (see (11)–(13)) without needing the instance values. In our experiments, we have removed $C_{\varphi_v}(\mathbf{x}_i^v)$ from the update rule (11) as if it is canceling out. This is done, as we wish to treat problems where only the pairwise distances are available for each view and not the instances themselves. In such cases, the dimensionality of the views is not known in order to compute $C_{\varphi_v}(\mathbf{x}_i^v)$. Such problems are very common in practice and we would like to test the proposed algorithm under this setting. Also, for our method a single execution of the EM algorithm has been always performed using a uniform initialization ($\pi^{v(0)} = 1/V, q_j^{(0)} = 1/N$), since no prior information for the quality of the views exists in any of the datasets.

Moreover, in each experiment the partition returned by all the aforementioned clustering methods is used to initialize an execution of the kernel k-means algorithm [29]. Such a run is conducted in order to determine whether there is room for improving CMMs results, or they are already close to a very good solution that cannot be further fine-tuned. A linear kernel, in order to be consistent with the choice of $d_{\varphi_v}(\mathbf{x}_i^v, \mathbf{x}_j^v) = \|\mathbf{x}_i^v - \mathbf{x}_j^v\|^2$, is selected for each view and since kernel k-means is a single-view method, a final kernel is built as a weighted sum of the individual view kernels. Those weights when fine-tuning our algorithm are the final π^v values, while for the multiview CMM they are set equal to $1/V$. For the single-view cases no weight is used. Note that for the concatenated case the linear kernel is calculated on the appended view vectors.

To further explore the potential of our method, we compare it to a multiview algorithm from the literature, which is built upon kernel CCA[2] [5]. This approach simultaneously uses all views to find appropriate projection directions that maximize the correlation between the projected views and then applies k-means to the projections of one of the views to get a partitioning of the instances. For the experiments with kernel CCA, we use the algorithm implementation made available by the authors of [5] and adopt a very similar experimental protocol as in [5]. Specifically, the number of projection axes is set equal to the cluster number, a linear kernel is selected (for the same reason as for kernel k-means), the kernel CCA regularization parameters are determined automatically using grid search (where kernel CCA is rerun with a new set of parameter values) and an appropriate criterion (see [5] for details), and k-means is restarted 30 times with random initializations and the run with the smallest k-means objective is kept. Note

[2]For simplicity, we shall refer to the clustering framework of [5] as kernel CCA in the experiments, although kernel CCA is only a part of this method.

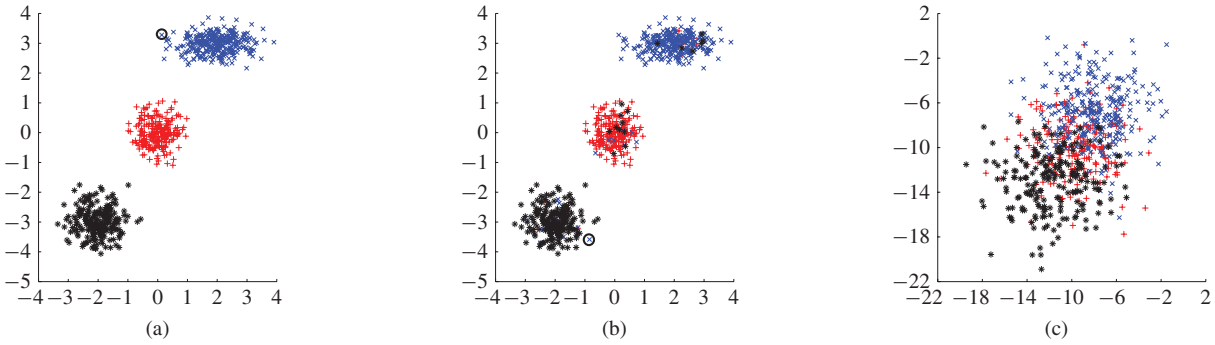

Fig. 1. Examples of the synthetic dataset: (a) original dataset generated from three 2-d Gaussian distributions, (b) one of the "corrupted" views for $m = 50$ and zero translation. The circled point in (a) (blue class) is wrongly represented here as belonging to the black class (circled point), and (c) one of the noisy views.

TABLE I

ARTIFICIAL-NOISE-FREE DATASETS RESULTS WITH GAUSSIAN CMM-BASED METHODS, IN TERMS OF ENTROPY AND THREE CLUSTERS. $m = 50$, $\beta^v = \beta_0^v$. THE "YES," "NO" COLUMNS INDICATE WHETHER KERNEL k-MEANS FINE TUNING IS APPLIED OR NOT. ALSO RESULTS WITH KERNEL CCA ARE REPORTED

	Two views		**Three views**		**Four views**		**Five views**	
	Kernel k-means		Kernel k-means		Kernel k-means		Kernel k-means	
	No	Yes	No	Yes	No	Yes	No	Yes
Worst single-view CMM	0.300	0.300	0.572	0.300	0.572	0.303	0.572	0.303
Best single-view CMM	0.299	**0.299**	0.299	0.299	0.299	0.299	0.299	0.299
Concatenated view CMM	0.322	0.320	0.265	**0.262**	0.147	**0.191**	0.130	**0.124**
Multiview CMM	**0.289**	0.320	**0.133**	**0.262**	0.097	0.195	0.081	**0.124**
Weighted multiview CMM	**0.289**	0.326	0.176	0.266	**0.086**	**0.191**	**0.060**	**0.124**
Kernel CCA—worst view	0.745		0.766		0.766		0.766	
Kernel CCA— best view	0.743		0.741		0.739		0.735	

TABLE II

INDICATIVE WEIGHTS ASSIGNED TO THE VIEWS BY THE WEIGHTED MULTIVIEW CMM FOR THE ARTIFICIAL-NOISE-FREE DATASETS

	Two views	**Three views**	**Four views**	**Five views**
View 1	0.502	0.336	0.251	0.201
View 2	0.498	0.333	0.249	0.199
View 3	-	0.331	0.248	0.198
View 4	-	-	0.252	0.201
View 5	-	-	-	0.201

that the grid search steps grow exponentially with the number of views and, together with the fact that kernel CCA requires solving a generalized eigenvalue problem (which is a timely procedure), make the application of this method prohibitive for datasets with many views. Also note, that since it is not clear which view's projections to use to get the final clustering with k-means, we cluster each of the available views' projections and report results for the best and worst performing ones.

For each dataset, the ground-truth class of every instance is available and the number of clusters is set equal to the true class number, unless stated otherwise. To assess the returned clusters quality, the *average entropy* metric [4], [7], [20], which measures the impurity of the partitions w.r.t. the ground truth classes, is used. Average entropy is given by (16), where N is the dataset size, M the number of clusters, c the number of classes, n_i^j the number of points in cluster i from class j, and n_i the size of the ith cluster. Lower average entropy values indicate that each cluster consists of instances belonging to the same class

$$H = \sum_{i=1}^{M} \frac{n_i}{N} \left(-\sum_{j=1}^{c} \frac{n_i^j}{n_i} \log \frac{n_i^j}{n_i} \right). \tag{16}$$

It must be emphasized that in all tested methods the ground-truth labels have not been used during training. They are used only to compute the performance measures after training.

B. Synthetic Datasets

The weighted multiview CMM is first tested on a dataset with 700 instances, generated from three 2-D Gaussian distributions (Fig. 1(a)). Each of the distributions represents a distinct class and this serves as the ground truth. From this original dataset, seven artificial views were created. For each of the first five views, as a first step all original instances were equally translated and then m of them were randomly selected and replaced by new ones. To replace each of the m instances, we randomly picked a class different from the one that the instance belongs to in the original dataset, and generated a new point from the corresponding class distribution. Therefore each view is "corrupted," as according to the ground truth m points belong to an incorrect class. For the experiments we set $m = 50$ and an example is illustrated in Fig. 1(b). For the

TABLE III

ARTIFICIAL-NOISY DATASETS RESULTS WITH GAUSSIAN CMM-BASED METHODS, IN TERMS OF ENTROPY AND THREE CLUSTERS. $m = 50$, $\beta^v = \beta_0^v$. THE "YES," "NO" COLUMNS INDICATE WHETHER KERNEL k-MEANS FINE TUNING IS APPLIED OR NOT. ALSO RESULTS WITH KERNEL CCA ARE REPORTED

	Two views + noisy views		**Three views + noisy views**		**Four views + noisy views**		**Five views + noisy views**	
	Kernel k-means		Kernel k-means		Kernel k-means		Kernel k-means	
	No	Yes	No	Yes	No	Yes	No	Yes
Worst single-view CMM	0.935	0.943	0.935	0.943	0.935	0.943	0.935	0.943
Best single-view CMM	0.299	0.299	0.299	0.299	0.299	0.299	0.299	0.299
Concatenated view CMM	0.960	0.748	0.477	0.631	0.472	0.496	0.377	0.402
Multiview CMM	0.385	0.751	0.655	0.631	0.147	0.478	0.576	0.601
Weighted multiview CMM	**0.256**	**0.281**	**0.220**	**0.242**	**0.127**	**0.206**	**0.116**	**0.157**
Kernel CCA—worst view	1.001		1.078		1.078		1.078	
Kernel CCA—best view	0.833		0.814		0.800		0.745	

TABLE IV

INDICATIVE WEIGHTS ASSIGNED TO THE VIEWS BY THE WEIGHTED MULTI-VIEW CMM, FOR THE ARTIFICIAL-NOISY DATASETS

	Two views + noisy views	**Three views + noisy views**	**Four views + noisy views**	**Five views + noisy views**
View 1	0.442	0.304	0.224	0.182
View 2	0.435	0.300	0.223	0.181
View 3	-	0.299	0.222	0.180
View 4	-	-	0.226	0.184
View 5	-	-	-	0.183
Noisy view 1	0.064	0.051	0.054	0.046
Noisy view 2	0.059	0.046	0.051	0.044

remaining two views, a high amount of zero-mean Gaussian noise was added to the original instances (noise std = 2.5), making it hard to separate the classes (Fig. 1(c)).

Individual clusterings of the five views will probably misclassify all m misplaced instances. We wish to examine whether the simultaneous consideration of multiple views helps to "fix" some of these errors. The noisy views contain little information for the problem and we want to explore how that fact is reflected by the weights π^v. Note that the original dataset is correctly separated by a CMM, i.e., $H = 0$.

Four noise-free datasets were constructed, including 2, 3, 4, and 5 "corrupted" views. Also, four noisy datasets were created by adding the two noisy views to the noise-free datasets. Results for the noise-free and noisy datasets are reported in Tables I and III, respectively, for three clusters and $\beta^v = \beta_0^v$ (8).

From Table I we see that the multiview methods (in the following Section V-B, when writing multiview methods we refer to the CMM-based ones and not kernel CCA) always outperform the best single view (apart from one case), indicating that multiple views contribute to the correction of the errors in the individual views. The concatenated view is always inferior or equal to at least one of the multiview approaches. When no kernel k-means fine-tuning is used, both multiview approaches are ahead. Therefore, appending the view vectors is not a good strategy, which is something widely mentioned in the literature (e.g., [7], [20]).

Moreover, from Table II we observe that our new algorithm roughly assigns the same weights to the views, which is something expected given that all views are of similar quality and which makes the method behave like the multiview CMM. This observation is in accordance with the results, where the two methods are of similar performance. Also, the multiview schemes take advantage of every available view, as the entropy constantly drops with the increase of the views number. Note that kernel k-means always degrades the performance of the multiview settings. Finally, kernel CCA[3] is systematically beaten by a large margin by all CMM-based methods and its performance barely increases as more views become available, highlighting the strength of CMMs.

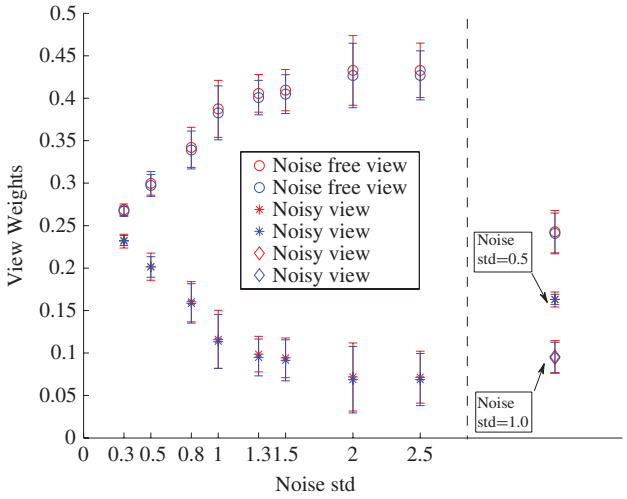

Fig. 2. View weights (average and std over 10 trials) assigned by the weighted multiview CMM to datasets consisting of two noise-free and two noisy views (both with the same amount of noise) for various noise levels. On the right, the weight averages and std for a dataset, where views with different amounts of noise simultaneously exist, are shown.

[3]We stress that in none of the experiments of this or the following sections we have applied kernel k-means to the partitions returned by kernel CCA.

TABLE V
WEBKB RESULTS WITH GAUSSIAN CMM-BASED METHODS FOR $\beta^v = \beta_0^v$ AND $\beta^v = \alpha^* \beta_0^v$, IN TERMS OF ENTROPY AND SIX CLUSTERS. THE "YES," "NO" COLUMNS INDICATE WHETHER KERNEL k-MEANS FINE TUNING IS APPLIED OR NOT. ALSO RESULTS WITH KERNEL CCA ARE REPORTED

	$\beta^v = \beta_0^v$		$\beta^v = \alpha^* \beta_0^v$	
	Kernel k-means		Kernel k-means	
	No	Yes	No	Yes
Single-view CMM—text	1.536	1.513	1.485 ($\alpha^* = 1.5$)	1.492 ($\alpha^* = 1.5$)
Single-view CMM—anchor text	1.554	1.471	1.440 ($\alpha^* = 3.5$)	1.329 ($\alpha^* = 3.5$)
Concatenated view CMM	1.559	1.537	1.481 ($\alpha^* = 1.7$)	1.490 ($\alpha^* = 1.7$)
Multiview CMM	1.498	1.450	1.396 ($\alpha^* = 1.5$)	1.316 ($\alpha^* = 1.5$)
Weighted multiview CMM	1.431	1.427	**1.299** ($\alpha^* = 3.5$)	**1.307** ($\alpha^* = 3.5$)
Kernel CCA—text	1.411			
Kernel CCA—anchor text	1.309			

When noise comes into play, the true potential of the proposed algorithm becomes apparent, since it achieves by far the least entropy in all cases. This happens because it *assigns very small weights to the noisy views* (as can be seen in Table IV), hence they are almost eliminated from the clustering process whereas the noise-free views are equally treated. Therefore, this method works as if the noise does not exist. Indeed, note that the weighted multiview CMM performance is relatively close to that of the noise-free setting. The advantages of automatically determining the view weights are now clearly exposed, as the method exhibits robustness to noise and to noninformative views in general.

In contrast, the multiview CMM splits are greatly affected by the noise, due to the equally weighted views, and are considerably inferior to the corresponding noise-free ones and even to those of the best single view (which is a noise-free view). Kernel CCA is also affected by the presence of noise, as the entropy has increased compared to that of Table I and is largely outperformed by the weighted multiview CMM, demonstrating the need for methods that distinguish noisy views. Once again, it is beaten by the multiview CMM and the best single view. Additionally, the time concerns regarding kernel CCA (see Section V-A) became evident when handling more than four views, when it took several hours to find the clusters, while our framework required a few minutes. Finally, similar conclusions as above can be drawn regarding the concatenated view and the application of kernel k-means.

To further investigate the behavior of the newly proposed algorithm, we studied the sensitivity of the weights on the noise level present on the views. Analytically, we created datasets for various amounts of noise which each consisted of two noise-free views (for all cases these are the ones used for the two-view experiment above) and two noisy views with the same amount of noise. To create the noisy views, random zero-mean Gaussian noise was added to the original dataset, with different standard deviation for the various noise levels. In order to alleviate randomness in our experiments, for each noise level we created 10 datasets and repeated the clustering. The average and standard deviation of the weights of the four views over 10 runs are depicted in Fig. 2. It can be seen that, as the noise increases, the difference among the noise-free views and noisy views weights becomes greater, which is something that we would naturally expect. Note that for std $= 0.3$, the weight values of the noisy views approach those of the noise-free ones as a low amount of noise is present, while for std $= 2.5$ the noisy views weights have a value around 0.06.

TABLE VI
INDICATIVE WEIGHTS ASSIGNED TO THE VIEWS BY THE WEIGHTED MULTIVIEW CMM FOR THE WEBKB DATASET WHEN $\beta^v = \beta_0^v$

	$\beta^v = \beta_0^v$
Text view	0.126
Anchor text view	0.874

Finally, a dataset that combines views with two different noise levels (two views for each noise level) and two noise-free views was constructed. The average and standard deviation of the weights over 10 trials are shown on the right-most corner of Fig. 2. We observe that the less informative a view is, the smaller its weight. Also, views with the same amount of noise are assigned very similar weights. From Fig. 2, we conclude that the new method identifies noisy views and treats them according to their noise level.

C. WebKb Dataset

The first real-world example is a popular collection for testing multiview algorithms [1], [3], [4], [7], [20] made up of the computer science department web pages from various universities. Here, the version described in [20] is used, consisting of six classes (course, department, faculty, project, staff, and students) and two views. The views are the text of the pages and the anchor text of all inbound links. As all web pages do not have inbound links, such instances were removed from the dataset, resulting in 2076 instances with both views available.

Term frequency inverse document frequency (tfidf) vectors were constructed for each view and normalized to unit length, so that the squared Euclidean distances of the Gaussian CMMs reflect the cosine similarity, which is usually employed to document clustering. The number of clusters was always set to six. Experiments with the CMM-based methods were performed for $\beta^v = \beta_0^v$ (8). We also considered other β^v values for each tested method, by setting $\beta^v = \alpha \beta_0^v$ and repeating the clustering for several α values. The value of

TABLE VII

INTAD RESULTS WITH GAUSSIAN CMM-BASED METHODS FOR $\beta^v = \beta_0^v$, IN TERMS OF ENTROPY AND DIFFERENT NUMBER OF CLUSTERS. THE "YES," "NO" COLUMNS INDICATE WHETHER KERNEL k-MEANS FINE TUNING IS APPLIED OR NOT. ALSO RESULTS WITH KERNEL CANONICAL CORRELATION ANALYSIS ARE REPORTED

	Two clusters		**Four clusters**		**Six clusters**	
	Kernel k-means		Kernel k-means		Kernel k-means	
	No	Yes	No	Yes	No	Yes
Worst single-view CMM	0.517	0.497	0.513	0.474	0.478	0.460
Best single-view CMM	**0.366**	0.382	**0.316**	0.314	0.366	0.344
Concatenated view CMM	0.496	0.496	0.489	0.489	0.472	0.474
Multiview CMM	0.481	0.362	0.393	**0.284**	0.393	0.290
Weighted multiview CMM	0.468	**0.349**	0.403	0.347	0.404	**0.267**
Kernel CCA—worst view	0.517		0.425		0.389	
Kernel CCA—best view	0.459		0.386		0.343	

TABLE VIII

INTAD RESULTS WITH GAUSSIAN CMM-BASED METHODS FOR $\beta^v = \alpha^* \beta_0^v$ IN TERMS OF ENTROPY AND DIFFERENT NUMBER OF CLUSTERS. THE "YES," "NO" COLUMNS INDICATE WHETHER KERNEL k-MEANS FINE TUNING IS APPLIED OR NOT. ALSO RESULTS WITH KERNEL CCA ARE REPORTED

	Two clusters		**Four clusters**		**Six clusters**	
	Kernel k-means		Kernel k-means		Kernel k-means	
	No	Yes	No	Yes	No	Yes
Worst single-view CMM	0.517 ($\alpha^* = 1$)	0.497 ($\alpha^* = 1$)	0.472 ($\alpha^* = 1$)	0.460 ($\alpha^* = 1$)	0.450 ($\alpha^* = 1.2$)	0.422 ($\alpha^* = 1.2$)
Best single-view CMM	0.366 ($\alpha^* = 1$)	0.382 ($\alpha^* = 1$)	**0.316** ($\alpha^* = 1$)	0.314 ($\alpha^* = 1$)	0.353 ($\alpha^* = 3$)	0.318 ($\alpha^* = 3$)
Concatenated view CMM	0.462 ($\alpha^* = 2$)	0.456 ($\alpha^* = 2$)	0.391 ($\alpha^* = 0.5$)	0.356 ($\alpha^* = 0.5$)	0.402 ($\alpha^* = 0.5$)	0.366 ($\alpha^* = 0.5$)
Multiview CMM	0.386 ($\alpha^* = 1.5$)	**0.288** ($\alpha^* = 1.5$)	0.362 ($\alpha^* = 1.2$)	0.324 ($\alpha^* = 1.2$)	0.339 ($\alpha^* = 1.2$)	0.283 ($\alpha^* = 1.2$)
Weighted multiview CMM	**0.337** ($\alpha^* = 3.5$)	0.299 ($\alpha^* = 3.5$)	0.357 ($\alpha^* = 3$)	**0.295** ($\alpha^* = 3$)	**0.331** ($\alpha^* = 1.2$)	**0.271** ($\alpha^* = 1.2$)
Kernel CCA—worst view	0.517		0.425		0.389	
Kernel CCA—best view	0.459		0.386		0.343	

α (denoted as α^*) that yielded the least entropy was selected as the best solution and its results are reported here. This was done in order to show that the results can be possibly improved by trying β^v values around β_0^v. Note that a common value α was used for all the views in the multiview algorithms. When applying kernel k-means, the β^v values already picked by the CMMs were retained. Obviously, for kernel CCA no β^v parameter exists for fine-tuning.

From Table V, it is apparent that the proposed clustering algorithm is beaten only by kernel CCA and only when $\beta^v = \beta_0^v$. It is superior though when β^v is fine-tuned, demonstrating that gains in performance are possible by searching around β_0^v and that it is a strong competitor amongst the literature multiview approaches. Compared to the multiview CMM, the gap in performance mainly emanates from the different view weights. Table VI contains the weights returned by the weighted multiview CMM when $\beta^v = \beta_0^v$, where a higher value is given to the anchor text view. Note that our algorithm achieves its best entropy (1.299) for the optimum β^v and with no kernel k-means postprocessing (fourth column). This is lower than the multiview CMM best (1.316), which is achieved for the optimum β^v with kernel k-means (fifth column). This indicates that for WebKb our method provides higher gains, without needing fine-tuning of the returned clusters. The multiview approaches are always ahead of the single views and the concatenated view, demonstrating once again the advantages of incorporating multiple views to the clustering task and the inefficiency of naive vector merging. The concatenated view in most cases is even worse than the single views. Finally, for $\beta^v = \beta_0^v$ kernel k-means improves the results, while for the best β^v it does so in half of the cases.

D. Internet Advertisement Dataset

The Internet advertisement dataset (Intad)[4] contains images from various web pages that are characterized either as advertisements or non-advertisements (i.e., two classes). The instances are described in terms of six views, which are the geometry of the images (width, height, aspect ratio), the phrases in the url of the pages containing the images (base url), the phrases of the images' url (image url), the phrases in the url of the pages the images are pointing at (target url), the anchor text, and the text of the images' alt (alternative) html tags (alt text). All views have binary features,

[4]Available from the UCI repository: http://www.ics.uci.edu/~mlearn/MLRepository.html

TABLE IX
INDICATIVE WEIGHTS ASSIGNED TO THE VIEWS BY THE WEIGHTED MULTIVIEW CMM FOR THE INTAD DATASET WHEN $\beta^v = \beta_0^v$

	Two clusters	Four clusters	Six clusters
Image url view	0.047	0.047	0.047
Base url view	0.237	0.237	0.237
Target url view	0.355	0.355	0.355
Alt text view	0.361	0.361	0.361

apart from the geometry view whose features are continuous. Details for the construction of the dataset can be found in [30]. Note that there are several missing views in this dataset. Specifically, the anchor text view is missing from 94% of the images and the geometry view from 30%, therefore we decided not to include those views in our empirical evaluation. After removing instances that were missing any of the other four views, 2369 images remained for the experiments.

Similarly to WebKb, we generated normalized tfidf vectors to reflect the cosine similarity and performed experiments both for $\beta^v = \beta_0^v$ (8) and $\beta^v = \alpha^* \beta_0^v$. Different cluster numbers were tried, specifically two, four and six.

The results in Tables VII and VIII[5] show that one of the two CMM-based multiview methods achieves the least entropy in most cases (in 8 out of 12). In the other cases, the best single view (two times for $\beta^v = \beta_0^v$ and one for $\beta^v = \alpha^* \beta_0^v$) and kernel CCA (only once, for six clusters and $\beta^v = \beta_0^v$) are superior. Note that both these multiview approaches are always ahead of the worst single view. Therefore, for the Intad dataset, if we test the views one by one, we might sometimes get a better partitioning than when simultaneously using all of them, particularly for $\beta^v = \beta_0^v$. The two aforementioned multiple view algorithms, though, provide higher quality solutions more systematically, especially if the β^v values are fine-tuned, since in Table VIII the best single view is ahead for only one setting. It is important to stress that kernel CCA is inferior to both the other two multiview approaches when $\beta^v = \alpha^* \beta_0^v$ and also for $\beta^v = \beta_0^v$, when kernel k-means is applied. Moreover, it is worse than the best single-view CMM most of the times (in 9 out of 12). This result shows that CMMs are a powerful clustering technique and support our decision to adopt them for a multiview framework.

Once again, the concatenated view is always inferior to the multiview schemes (only kernel CCA performs worse in a few cases) and also to the best single view (sometimes even to the worst single view). The proposed algorithm is superior to multiview CMMs for 8 out of 12 cases and is the best overall performer 6 times out of 12, contrary to the multiview CMMs two times. The difference among the two methods is greater in Table VIII, where the weighted multiview CMM is ahead for all but one case. As an indication of how our algorithm handles this dataset, in Table IX the view weights for the run with $\beta^v = \beta_0^v$ are given. Note that their value is the same for all cluster numbers, since it does not depend on this parameter. Finally, the application of kernel k-means seems to be beneficial for all cases.

[5]The values reported for kernel CCA are the same in both tables, since for kernel CCA no β^v parameter to fine tune exists.

VI. CONCLUSION AND FUTURE WORK

We have introduced the weighted multiview CMM, which is a weighted combination of exemplar-based models that identifies exemplars in the dataset by simultaneously considering multiple representations of the instances. Our method can be interpreted as a mixture model, whose components are CMMs (one for each view). Its main advantages are the assignment of different weights to the views, which are automatically determined and provide robustness against noisy or low quality views, as well as the ability to handle views with different statistical properties. Also, it is computationally efficient and involves simple iterative calculations when optimizing the parameters, using the well-known EM procedure. As only priors take part in the maximization of the likelihood, a uniform initialization is the most natural choice in the absence of *a priori* knowledge about the importance of the views. A single, uniformly initialized execution of EM was adopted throughout the experimental evaluation and, based on the satisfactory experimental results obtained for all datasets, we can safely claim that multiple restarts for EM can be avoided. Finally, the employment of the method for multiview feature space clustering is straightforward.

The proposed framework has been tested on several diverse datasets and compared with the multiview CMM [10] as well as with the single-view CMM [12] (applied to each individual view and the concatenated view) and also with the method of [5] that clusters the projections obtained by kernel CCA across the views. In general, the results verify the superiority of the weighted multiview CMM. Its performance is constantly the best for the noisy versions of the synthetic datasets. When no noise is present in the synthetic data, it is matched only by the multiview CMM. This is expected, as all views are approximately of the same quality and therefore the impact of weights is minimal. Also, experiments with varying amount of noise on the views showed that the weights assigned to them are in direct association with their noise level. For the real datasets, our method is ahead for most of the cases, especially for $\beta^v = \alpha^* \beta_0^v$. Importantly, it has proven superior to the kernel CCA-based clustering scheme [5] in general (it is only worse for WebKb when $\beta^v = \beta_0^v$ and for Intad when $\beta^v = \beta_0^v$ and kernel k-means is not applied). All the above indicate that our algorithm is a worthy addition to the existing literature.

Overall, the experiments have shown that multiple views are beneficial in identifying good partitions, particularly if the views participate with different weights. Also, the proposed algorithm produces robust high-quality clusters under different settings. Moreover, it has been demonstrated that the concatenation of the representations is not an effective strategy and that the success of fine-tuning the solutions of the CMM-based methods with kernel k-means is dataset-dependent.

Before concluding this paper, we provide some more discussion regarding the usefulness of multiple views and the applicability of our method. A typical setting where we gain advantage from our multiview approach is when there is a majority of views suggesting the same clustering structure, but the single-view solutions are inferior because the views are corrupted due to noise or due to some other reason. In

this case, the proposed multiview algorithm is able to recover the correct clustering structure by combining complementary information from many views (this is the artificial experiment described in Section V-B). Of course if the corruption in each single view is not random, and thus the differences between views exhibit a systematic bias, then there is not sufficient information to recover the correct clustering structure. Hence, combining data from multiple views may not be always beneficial. Another case where our multiview method seems to be effective is when there are irrelevant views suggesting no significant clustering structure. In such a case, the algorithm seems to be able identify such views and assign low weight to them. From another viewpoint, the proposed method can be considered as providing a ranking of the individual views through the π^v values. This ranking can be further exploited, for example by keeping only the top views (even the best view only) and applying again the clustering algorithm.

An interesting method that attempts to infer the dependence relationship between two views is presented in [31] for clustering RNA data using both proteomic and transcriptomic expression profiles. In this approach, each view is modeled using a different mixture model, but the two mixture models are "coupled" in the sense that we define a joint prior distribution over both sets of components. This joint prior distribution allows for solutions ranging from the completely independent to the totally dependent case and is inferred from the data when training the mixture models. It is clear that our method in its current form is not able to explicitly model the dependence relationship between the views. However, it is not clear whether the coupled mixture model approach mentioned previously can be used for more than two views.

In our future paper, we plan to thoroughly investigate the application of the weighted multiview CMM into multiview feature space clustering and compare it to other multiview approaches. In addition, fine-tuning the returned clusters with techniques other than kernel k-means is in our plans. Furthermore, we believe that the idea of weighted views can be further exploited in different ways, either by inventing new algorithms or by modifying existing ones. Another interesting research direction is the development of learning frameworks that simultaneously perform clustering and classification in such a way that these two tasks complement each other, as suggested in [32], in the case of multiply represented data. Finally, cluster extraction for multiview instances using self-organizing maps (SOMs) is in our plans, for example by exploiting the SOMs' knowledge of the data topology as in [33].

APPENDIX A
PROOF OF THE EM ALGORITHM FOR WEIGHTED MULTIVIEW CMMS

For clarity, we will restate here all mathematical quantities that are necessary for our proof and explicitly declare the parameters they depend upon. Given a dataset $\mathfrak{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_i = \left\{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^V\right\}$, $\mathbf{x}_i^v \in \Re^{d^v}$ the distribution of our mixture model is

$$F(\boldsymbol{x}; \boldsymbol{\Theta}) = \sum_{v=1}^{V} \pi^v Q^v\left(\mathbf{x}^v; \left\{q_j\right\}_{j=1}^N\right), \ \mathbf{x}^v \in \Re^{d^v} \tag{17}$$

where

$$Q^v(\mathbf{x}^v; \left\{q_j\right\}_{j=1}^N) = \sum_{j=1}^{N} q_j f_j^v(\mathbf{x}^v), \boldsymbol{\Theta} = \left\{\left\{\pi^v\right\}_{v=1}^V, \left\{q_j\right\}_{j=1}^N\right\}$$

$$\pi^v \geq 0, \ \sum_{v=1}^{V} \pi^v = 1, \ q_j \geq 0, \ \sum_{j=1}^{N} q_j = 1.$$

Note that the exponential family distributions $f_j^v(\mathbf{x}^v)$ are independent of the parameters $\boldsymbol{\Theta}$. Our target is to maximize the log-likelihood (18) of the dataset $\mathfrak{X}$ under the mixture model distribution $F(\boldsymbol{x}; \boldsymbol{\Theta})$, w.r.t. the parameters $\boldsymbol{\Theta}$

$$L(\mathfrak{X}; \boldsymbol{\Theta}) = \sum_{i=1}^{N} \log \sum_{v=1}^{V} \pi^v Q^v(\mathbf{x}_i^v; \left\{q_j\right\}_{j=1}^N). \tag{18}$$

Let $\{\mathfrak{X}, \mathcal{Z}\}$ be the complete dataset, where $\mathcal{Z} = \{z_1, z_2, \ldots, z_N\}$ contains the latent variables indicating the mixture component responsible for generating each instance, i.e., $z_i \in \{1, 2, \ldots, V\}$. It must be stressed that the z_i values are not known in practice. Now we will analytically prove each step of the EM process [28].

From our model the next probabilities directly follow

$$P(z_i = v; \boldsymbol{\Theta}) = \pi^v \tag{19}$$

$$P(\boldsymbol{x}_i | z_i = v; \boldsymbol{\Theta}) = Q^v\left(\mathbf{x}_i^v; \left\{q_j\right\}_{j=1}^N\right). \tag{20}$$

E-step: This step uses the current parameter values $\boldsymbol{\Theta}^{(t)}$ to find the posteriors of the latent variables $P(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)})$, by applying the Bayes' theorem

$$P\left(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right) = \frac{P\left(z_i = v; \boldsymbol{\Theta}^{(t)}\right) P\left(\boldsymbol{x}_i | z_i = v; \boldsymbol{\Theta}^{(t)}\right)}{P\left(\boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right)} = \frac{\pi^{v(t)} Q^v\left(\mathbf{x}_i^v; \{q_j^{(t)}\}_{j=1}^N\right)}{\sum_{v=1}^V \pi^{v(t)} Q^v\left(\mathbf{x}_i^v; \{q_j^{(t)}\}_{j=1}^N\right)}. \tag{21}$$

M-step: This step calculates the expectation of the complete dataset log-likelihood (22) under the current latent variables posterior distribution, evaluated for some general parameter value $\boldsymbol{\Theta}$

$$\begin{aligned}
\mathcal{Q}(\boldsymbol{\Theta}; \boldsymbol{\Theta}^{(t)}) &= \sum_{i=1}^{N} < \log P(z_i, \boldsymbol{x}_i; \boldsymbol{\Theta}) >_{P\left(z_i | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right)} \\
&= \sum_{i=1}^{N} \sum_{v=1}^{V} \Big[P\left(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right) \\
&\quad \times \log\left(P(z_i = v; \boldsymbol{\Theta}) P(\boldsymbol{x}_i | z_i = v; \boldsymbol{\Theta})\right) \Big] \\
&= \sum_{i=1}^{N} \sum_{v=1}^{V} \Big[P\left(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right) \\
&\quad \times \log\left(\pi^v Q^v\left(\mathbf{x}_i^v; \left\{q_j\right\}_{j=1}^N\right)\right) \Big] \\
&= \sum_{i=1}^{N} \sum_{v=1}^{V} P\left(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right) \log \pi^v \\
&\quad + \sum_{i=1}^{N} \sum_{v=1}^{V} P\left(z_i = v | \boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}\right) \log\left(\sum_{j=1}^{N} q_j f_j^v(\mathbf{x}_i^v)\right).
\end{aligned} \tag{22}$$

Subsequently, it maximizes this expectation to get a new estimate for the parameters, taking into account any imposed constraints. The constrained expectation is given by (23), where λ and μ are Lagrange multipliers

$$\mathcal{Q}_{con}\left(\mathbf{\Theta};\mathbf{\Theta}^{(t)}\right)=\mathcal{Q}\left(\mathbf{\Theta};\mathbf{\Theta}^{(t)}\right)+\lambda\left(\sum_{v=1}^{V}\pi^{v}-1\right)+\mu\left(\sum_{j=1}^{N}q_j-1\right). \tag{23}$$

We begin by optimizing (23) w.r.t. π^v, by setting the corresponding derivative equal to zero

$$\frac{\partial\mathcal{Q}_{con}\left(\mathbf{\Theta};\mathbf{\Theta}^{(t)}\right)}{\partial\pi^{v}}=0\Rightarrow\sum_{i=1}^{N}\frac{P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)}{\pi^{v}}+\lambda=0$$
$$\Rightarrow-\lambda\pi^{v}=\sum_{i=1}^{N}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right). \tag{24}$$

Summing over v and making use of the constraint $\sum_{v=1}^{V}\pi^{v}=1$, we obtain

$$-\lambda\underbrace{\sum_{v=1}^{V}\pi^{v}}_{=1}=\sum_{i=1}^{N}\underbrace{\sum_{v=1}^{V}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)}_{=1}\Rightarrow\lambda=-N. \tag{25}$$

Using (25) to eliminate λ in (24) and rearranging gives the new estimation

$$\pi^{v(t+1)}=\frac{1}{N}\sum_{i=1}^{N}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right).$$

The maximization of (23) w.r.t. q_j, by setting the corresponding derivative equal to zero, follows:

$$\frac{\partial\mathcal{Q}_{con}\left(\mathbf{\Theta};\mathbf{\Theta}^{(t)}\right)}{\partial q_j}=0$$
$$\Rightarrow\sum_{i=1}^{N}\sum_{v=1}^{V}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)\frac{f_j^{v}(\mathbf{x}_i^{v})}{\sum_{j'=1}^{N}q_{j'}f_{j'}^{v}(\mathbf{x}_i^{v})}+\mu=0. \tag{26}$$

By multiplying both sides of (26) with q_j and then summing over j, together with the constraint $\sum_{j=1}^{N}q_j=1$, gives

$$-\mu\underbrace{\sum_{j=1}^{N}q_j}_{=1}=\underbrace{\sum_{i=1}^{N}\sum_{v=1}^{V}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)\underbrace{\sum_{j=1}^{N}\frac{q_j f_j^{v}(\mathbf{x}_i^{v})}{\sum_{j'=1}^{N}q_{j'}f_{j'}^{v}(\mathbf{x}_i^{v})}}_{=1}}_{=1}$$
$$\Rightarrow\mu=-N. \tag{27}$$

Using (27) to eliminate μ in (26) and rearranging, we get

$$q_j=\frac{q_j}{N}\sum_{i=1}^{N}\sum_{v=1}^{V}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)\frac{f_j^{v}(\mathbf{x}_i^{v})}{\sum_{j'=1}^{N}q_{j'}f_{j'}^{v}(\mathbf{x}_i^{v})}. \tag{28}$$

Since we cannot solve analytically for q_j in (28), we must resort into iteratively performing updates on q_j during the M-step, before we proceed to the next EM iteration. That is the reason for writing t' for the new estimations in the following equation, instead of t

$$q_j^{(t'+1)}=\frac{q_j^{(t')}}{N}\sum_{i=1}^{N}\sum_{v=1}^{V}P\left(z_i=v|\boldsymbol{x}_i;\mathbf{\Theta}^{(t)}\right)\frac{f_j^{v}(\mathbf{x}_i^{v})}{\sum_{j'=1}^{N}q_{j'}^{(t')}f_{j'}^{v}(\mathbf{x}_i^{v})}.$$

Appendix B
Proof of the Assignment Step for Weighted Multiview CMMs

The weighted multiview CMM represents all instances as possible exemplars, since each view's CMM has N components, centered at the corresponding instances. Viewing our model as a mixture model, for a given parameter value $\mathbf{\Theta}$ an instance $\boldsymbol{x}_i$ is softly assigned to the jth component (cluster) with probability $P(c_i=j|\boldsymbol{x}_i;\mathbf{\Theta})$, where $c_i\in\{1,2,\ldots,N\}$ indicates the CMM component responsible for generating $\boldsymbol{x}_i$. Apparently, the value for c_i is unknown in practice. By applying Bayes' theorem, we write

$$P(c_i=j|\boldsymbol{x}_i;\mathbf{\Theta})=\frac{P(c_i=j;\mathbf{\Theta})P(\boldsymbol{x}_i|c_i=j;\mathbf{\Theta})}{P(\boldsymbol{x}_i;\mathbf{\Theta})}=\frac{P(c_i=j;\mathbf{\Theta})P(\boldsymbol{x}_i|c_i=j;\mathbf{\Theta})}{\sum_{v=1}^{V}\pi^{v}Q^{v}\left(\mathbf{x}_i^{v};\{q_j\}_{j=1}^{N}\right)}. \tag{29}$$

For our model it holds that

$$P(z_i=v|c_i=j;\mathbf{\Theta})=P(z_i=v;\mathbf{\Theta})=\pi^{v}, \tag{30}$$

$$P(c_i=j;\mathbf{\Theta})=P(c_i=j|z_i=v;\mathbf{\Theta})=q_j, \tag{31}$$

$$P(\boldsymbol{x}_i|z_i=v,c_i=j;\mathbf{\Theta})=f_j^{v}(\mathbf{x}_i^{v}). \tag{32}$$

The second nominator term with the help of (30) and (32) is estimated as

$$P(\boldsymbol{x}_i|c_i=j;\mathbf{\Theta})=\sum_{v=1}^{V}[P(\boldsymbol{x}_i|z_i=v,c_i=j;\mathbf{\Theta})\times P(z_i=v|c_i=j;\mathbf{\Theta})]=\sum_{v=1}^{V}\pi^{v}f_j^{v}\left(\mathbf{x}_i^{v}\right). \tag{33}$$

Substituting (31) and (33) into (29), we obtain

$$P(c_i=j|\boldsymbol{x}_i;\mathbf{\Theta})=\frac{q_j\sum_{v=1}^{V}\pi^{v}f_j^{v}\left(\mathbf{x}_i^{v}\right)}{\sum_{v=1}^{V}\pi^{v}Q^{v}\left(\mathbf{x}_i^{v};\{q_j\}_{j=1}^{N}\right)}. \tag{34}$$

Note that the above probability is used in (14) for those components whose corresponding instances are selected as the exemplars that are representing each of the M clusters.

Acknowledgment

The authors would like to thank S. Bickel and T. Scheffer for kindly providing their processed version of the WebKb dataset.

REFERENCES

[1] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. 11th Annu. Conf. Comput. Learn. Theory*, Madison, WI, 1998, pp. 92–100.

[2] U. Brefeld and T. Scheffer, "Co-EM support vector learning," in *Proc. 21st Int. Conf. Mach. Learn.*, Banff, AB, Canada, 2004, p. 16.

[3] I. Muslea, S. Minton, and C. A. Knoblock, "Active + semi-supervised learning = robust multi-view learning," in *Proc. 19th Int. Conf. Mach. Learn.*, Marina del Rey, CA, 2002, pp. 435–442.

[4] S. Bickel and T. Scheffer, "Multi-view clustering," in *Proc. 4th IEEE Int. Conf. Data Mining*, Washington D.C., Nov. 2004, pp. 19–26.

[5] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Anchorage, AK, Jun. 2008, pp. 1–8.

[6] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Montreal, QC, Canada, 2009, pp. 129–136.

[7] V. R. de Sa, "Spectral clustering with two views," in *Proc. 22nd Int. Conf. Mach. Learn. Workshop Learn. Multiple Views*, Bonn, Germany, 2005, pp. 20–27.

[8] D. Greene and P. Cunningham, "A matrix factorization approach for integrating multiple data views," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, Bled, Slovenia, 2009, pp. 423–438.

[9] B. Long, P. S. Yu, and Z. M. Zhang, "A general model for multiple view unsupervised learning," in *Proc. 8th SIAM Int. Conf. Data Mining*, Atlanta, GA, 2008, pp. 822–833.

[10] G. Tzortzis and A. Likas, "Convex mixture models for multi-view clustering," in *Proc. 19th Int. Conf. Artif. Neural Netw. Part II*, Limassol, Cyprus, 2009, pp. 205–214.

[11] D. Zhou and C. J. C. Burges, "Spectral clustering and transductive learning with multiple views," in *Proc. 24th Int. Conf. Mach. Learn.*, Corvalis, OR, 2007, pp. 1159–1166.

[12] D. Lashkari and P. Golland, "Convex clustering with exemplar-based models," in *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008, pp. 825–832.

[13] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proc. 33rd Annu. Meeting Assoc. Comput. Linguistics*, Cambridge, MA, 1995, pp. 189–196.

[14] M. Collins and Y. Singer, "Unsupervised models for named entity classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Florham Park, NJ, 1999, pp. 100–110.

[15] Z.-H. Zhou and M. Li, "Semi-supervised regression with co-training," in *Proc. 19th Int. Joint Conf. Artif. Intell.*, Edinburgh, Scotland, 2005, pp. 908–913.

[16] S. P. Abney, "Bootstrapping," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics*, Philadelphia, PA, 2002, pp. 360–367.

[17] S. Dasgupta, M. L. Littman, and D. A. McAllester, "PAC generalization bounds for co-training," in *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press, 2001, pp. 375–382.

[18] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *Proc. 9th Int. Conf. Inform. Knowl. Manage.*, McLean, VA, 2000, pp. 86–93.

[19] R. Ghani, "Combining labeled and unlabeled data for multiclass text categorization," in *Proc. 19th Int. Conf. Mach. Learn.*, Sydney, Australia, 2002, pp. 187–194.

[20] S. Bickel and T. Scheffer, "Estimation of mixture models using co-EM," in *Proc. 16th Eur. Conf. Mach. Learn.*, Bonn, Germany, 2005, pp. 35–46.

[21] V. R. de Sa, "Learning classification with unlabeled data," in *Advances in Neural Information Processing Systems 6*. San Mateo, CA: Morgan Kaufmann, 1994, pp. 112–119.

[22] V. R. de Sa and D. H. Ballard, "Category learning through multimodality sensing," *Neural Comput.*, vol. 10, no. 5, pp. 1097–1117, Jul. 1998.

[23] H. Valizadegan and R. Jin, "Generalized maximum margin clustering and unsupervised kernel learning," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2006, pp. 1417–1424.

[24] F. Wang, B. Zhao, and C. Zhang, "Linear time maximum margin clustering," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 319–332, Feb. 2010.

[25] B. Zhao, J. T. Kwok, and C. Zhang, "Multiple kernel clustering," in *Proc. 9th SIAM Int. Conf. Data Mining*, Sparks, NV, 2009, pp. 638–649.

[26] H. Zeng and Y.-M. Cheung, "Kernel learning for local learning based clustering," in *Proc. 19th Int. Conf. Artif. Neural Netw. Part I*, Limassol, Cyprus, 2009, pp. 10–19.

[27] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with Bregman divergences," *J. Mach. Learn. Res.*, vol. 6, pp. 1705–1749, Dec. 2005.

[28] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[29] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[30] N. Kushmerick, "Learning to remove internet advertisements," in *Proc. 3rd Annu. Conf. Autonomous Agents*, Seattle, WA, 1999, pp. 175–181.

[31] S. Rogers, M. Girolami, W. Kolch, K. M. Waters, T. Liu, B. Thrall, and H. S. Wiley, "Investigating the correspondence between transcriptomic and proteomic expression profiles using coupled cluster models," *Bioinformatics*, vol. 24, no. 24, pp. 2894–2900, 2008.

[32] W. Cai, S. Chen, and D. Zhang, "A multiobjective simultaneous learning framework for clustering and classification," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 185–200, Feb. 2010.

[33] K. Tasdemir and E. Merenyi, "Exploiting data topology in visualization and clustering of self-organizing maps," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 549–562, Apr. 2009.

Grigorios F. Tzortzis received the B.Sc. and M.Sc. degrees in computer science from the University of Ioannina, Ioannina, Greece, in 2006 and 2008, respectively. Currently, he is pursuing the Ph.D. degree in the Department of Computer Science, University of Ioannina.

His current research interests include machine learning, neural networks, multiview learning, and data mining.

Aristidis C. Likas (SM'03) received the Diploma degree in electrical engineering and the Ph.D. degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 1990 and 1994, respectively.

He has been with the Department of Computer Science, University of Ioannina, Ioannina, Greece, since 1996, where he is currently an Associate Professor. His current research interests include neural networks, machine learning, statistical signal processing, and bioinformatics.