Variational graph autoencoders for multiview canonical correlation analysis

Yacouba Kaloga, Pierre Borgnat, Sundeep Prabhakar Chepuri, Patrice Abry, Amaury Habrard, Sundeep Prabhakar Chepuri

Variational Graph Autoencoders for Multiview Canonical Correlation Analysis

Yacouba Kaloga[§], Pierre Borgnat[§], Sundeep Prabhakar Chepuri[⋆], Patrice Abry[§] and Amaury Habrard[† 1]

[§] *Univ Lyon, Ens de Lyon, Univ. Claude Bernard, CNRS, Laboratoire de Physique, Lyon, France*

[⋆] *Department of Electrical and Communication Engineering, Indian Institute of Science, Bangalore, India*

[†] *University of Lyon, UJM-Saint-Etienne, CNRS, Laboratoire Hubert Curien, UMR 5516, France*

Abstract

We present a novel approach for multiview canonical correlation analysis based on a variational graph neural network model. We propose a non-linear model which takes into account the available graph-based geometric constraints while being scalable to large-scale datasets with multiple views. This model combines the probabilistic interpretation of CCA with an autoencoder architecture based on graph convolutional neural network layers. Experiments with the proposed method are conducted on classification, clustering, and recommendation tasks on real datasets. The algorithm is competitive with state-of-the-art multiview representation learning techniques, in addition to being scalable and robust to instances with missing views.

Keywords: Canonical correlation analysis, Dimensionality reduction, Multiview representation learning, Graph neurals networks, Variational inference

[1]Supported by the IFCAM project MA/IFCAM/19/56, the ACADEMICS Grant of IDEXLYON, Univ. Lyon, PIA ANR-16-IDEX-0005, the ANR project DataRedux (ANR-19-CE46-0008), and the CBP IT test platform (ENS de Lyon, France) for ML facilities and GPU devices, operating the SIDUS solution [1].
Preliminary results were presented at the conference [2].

1. Introduction

Interconnected societies generate large amounts of structured data that frequently stem from observing a common set of objects (or sources) through different modalities. Such multiview datasets are also encountered in many different fields like computational biology [3], acoustics [4], surveillance [5], or social networks [6], to list a few. In many of these applications, datasets are structured (in graphs, trees or sequences), large and it is common that some of the views have missing entries. Although there exist many tools to analyze and study multiview datasets [7], analyzing large-scale structured multiview datasets with missing or incomplete views efficiently is still a very challenging task.

Canonical Correlation Analysis (CCA) [8, 9] can be used for multiview representation learning, by seeking latent low-dimensional representations that are common to the different views. This common representation that encodes information from different datasets can be leveraged to improve the performance of machine learning tasks, e.g., clustering [10]. There are two general approaches to CCA: *algebraic* or *probabilistic*.

The *algebraic* approaches to CCA were initially proposed for two-view data following [8] and they obtain a latent low-dimensional manifold by maximizing correlations between the projections of the different views onto it. Being nonparametric, these approaches are powerful and versatile but do not scale well to large datasets. Nevertheless, there have been numerous extensions: to the multiview setting, see [9]; or to account for nonlinear dependencies (beyond correlations), see Kernel CCA [11, 12], Deep CCA [13, 14], or Autoencoder CCA [15]. Despite significant improvements in performance, many of these approaches suffer from scalability issues [16, 17], mainly due to the prohibitive costs of the underlying eigendecomposition on which most of such methods rely on, and the difficulty to extent this settings beyond two views.

Alternatively, *probabilistic* approaches to CCA were developed: CCA solves a Bayesian inference problem [18]. As recent advances in variational autoencoders [19] made Bayesian inference scalable, the probabilistic CCA approaches gained popularity because of their potential (e.g., inference task such as generating new dataset samples) and scalability, e.g, see VCCA(p) [20], or VPCCA [21]. Using a probabilistic model, these methods scale easily to large datasets. However, they are less versatile to adjust to model mismatch or to data structure as these methods are model based.

Concomitantly to these advances, it was shown in [22, 23] that incorporating the available graph-induced knowledge about the common source into multiview CCA improves performance of various machine learning tasks. We refer to this graph-aware multiview CCA method from [23] as GMCCA. However, GMCCA suffers from the involved eigendecomposition costs. In essence, there are no CCA methods that have the advantages of both worlds: being able to incorporate prior graph-based structure in the latent space and being scalable. Such a method is proposed here. The present work attempts to reconcile scalability and versatility for multiview CCA.

In the following, we develop a scalable multiview variational graph autoencoder for CCA (MVGCCA), which is robust to the presence of instances with missing views in multiview datasets. Section 2 recalls some background and technical elements for multiview CCA. Section 3 describes the proposed approach and its key contributions. In particular, we show how graph structure can be enforced in the common latent space while preserving scalability. Additionally, we discuss how the proposed method is robust toexistence of instances with missing views, and how to improve that with the idea of "views dropout". Section 4 describes the datasets that are used for numerical experiments. These experiments are described and discussed in Section 5. Finally, we conclude in Section 6.

2. Multiview CCA

Multiview datasets - We consider M-view data X where each instance[2] has M views, each in space $\mathbb{R}^{d_m}$, $m = 1, \ldots, M$. We have n instances in X. The m-th view of instance i is written as $X_m^i \in \mathbb{R}^{d_m}$, and the collection of views for instance i is denoted as $X^i = \{X_m^i\}_{m=1}^M$. We also introduce the data matrix $X_m \in \mathbb{R}^{d_m \times n}$ related the m-th view, whose columns are X_m^i, $1 \leq i \leq n$. See Figure 1 for an illustration. Not to be confused with data $X_m^i \in \mathbb{R}^{d_m}$, we denote x_m any variable vector in $\mathbb{R}^{d_m}$, when needed in the text.

Graph - We assume that each instance is associated to a node in a graph $\mathcal{G}$ having a structure connecting the different instances. This graph captures closeness and similarities between the different instances. The adjacency matrix of the graph $\mathcal{G}$ is denoted A. We denote the neighborhood of node

[2]The instances can be designed as sources of the views.

i in the graph as $\mathcal{V}(i)$ and denote $\mathcal{V}^l(i)$ as the l hop neighborhood of node. We define also the following sets of features on the nodes (which are the views of instances) associated to these neighborhoods in the following way: Let us write $d^A_{i,j}$ the length of the shortest path in graph $\mathcal{G}$ between nodes (instances) i and j, and define:

$$\mathcal{V}^l(X^i) = \{X^j \in X | d^A_{i,j} \leq l\}$$

which is the set of multiview features of the neighborhood of i up to a distance l in graph $\mathcal{G}$ (of adjacency matrix A). We define the equivalent set, limited to view m:

$$\mathcal{V}^l(X^i_m) = \{X^j_m \in X_m | d_{i,j} \leq l\}$$

Notations - For any matrix B, $B(i,:)$ denotes the i-th row and $B(:,i)$ denotes the i-th column. The vector $[c_1, c_2, \ldots, c_p]^T$ obtained by concatenation is denoted by $[c_k]_{k=1}^p$. $\|\cdot\|_F$ is the Frobenius norm; $\mathrm{tr}(\cdot)$ is the trace operator; $[\![i,k]\!]$ is the set of integers between i and k (including the boundary). For given distributions p and q, $D_{KL}(p\|q)$ is the Kullback-Leibler distance between these distributions.

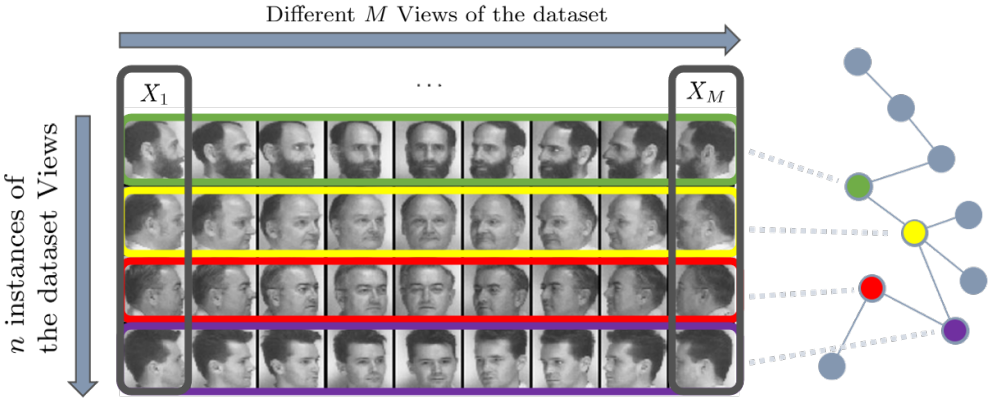


Figure 1: **Multiview dataset with graph structure.** An illustration of a multiview dataset. We have pictures taken from different angles for several people. The set of pictures of one subject is an instance and the set of pictures taken from the same angle is a view. Instances of the dataset can be related by an relationship on a graph. For example here, theses instances (i.e subjects) could be part of a social network, in which people are connected according to their friendship relations.

Method	Complexity	Non Linear	>2 views	Graph	Robustness
CCA	$O(n)$	✗	✗	✗	✗
Kernel CCA	$O(n^2)^3$	✓	✗	✗	✗
Deep CCA	$O(n)$	✓	✗	✗	✗
GMCCA	$O(n^2)$	✗	✓	✓	✗
VCCA(p)	$O(n)$	✓	✗	✗	✗
VPCCA	$O(n)$	✗	✓	✗	✗
MVGCCA	$O(n)$	✓	✓	✓	✓

Table 1: Key properties of methods related to CCA. n is the number of elements in the dataset. The column entitle 'Graph' indicates whether or not potential graph structure is taken into account. The column 'Robustness' indicates whether or not the model is robust to missing views in the data.

2.1. Algebraic approaches: linear CCA and extension

CCA. Let $X_1 \in \mathbb{R}^{d_1 \times n}$ and $X_2 \in \mathbb{R}^{d_2 \times n}$ denote two views of dimension d_1 and d_2 for n instances. Given a dimension $d \ll \min(d_1, d_2)$, CCA seeks the best projectors $U_1 \in \mathbb{R}^{d_1 \times d}$ and $U_2 \in \mathbb{R}^{d_2 \times d}$ such that the correlation between $U_1^T X_1$ and $U_2^T X_2$ is maximized. This can be formulated as the following optimization problem:

$$\min_{U_1, U_2} \|U_1^T X_1 - U_2^T X_2\|_F^2 \quad \text{s.t.} \quad U_m^T (X_m X_m^T) U_m = I_{d_m} \text{ for } m \in \{1, 2\}. \tag{1}$$

Introducing Σ_{11}, Σ_{22} the (regularized) correlation matrices[4]:

$$\Sigma_{mm} = \frac{1}{n-1} X_m X_m^T + r_m I_{d_m} \quad (r_m > 0) \quad (m = 1, 2) \tag{2}$$

and the cross correlation matrix $\Sigma_{12} = \frac{1}{n-1} X_1 X_2^T$. The solution (U_1^*, U_2^*) of the problem to Eq. (1) is obtained via an eigendecomposition [13, 23]:

$$(U_1^*, U_2^*) = ({\Sigma_{11}}^{-\frac{1}{2}} U_d, {\Sigma_{22}}^{-\frac{1}{2}} V_d) \tag{3}$$

[4] r_1 and r_2 are regularization parameters allowing to avoid degenerate correlation matrices and irrelevant correlations [24].

where $U_d \in \mathbb{R}^{d_1 \times d}$ (resp. $V_d \in \mathbb{R}^{d_2 \times d}$) are the d leading left (resp. right) eigenvectors of the matrix $T = {\Sigma_{11}}^{-\frac{1}{2}} \Sigma_{12} {\Sigma_{22}}^{-\frac{1}{2}}$. This formulation can easily be extended to take into account non-linear relationships by replacing, in Eq. (1), X_1 and X_2 by any non-linear function of X_1 and X_2 (e.g., Kernel CCA [11, 12], Deep CCA [13, 14], autoencoder CCA [15], etc.).

Multiview CCA (MCCA). A direct way to extend classical CCA to multiview datasets with $M > 2$ is to maximise pairwise correlations between all pairs of views:

$$\min_{(U_m)_{m=1}^M} \sum_{\substack{m=1 \\ m'>m}}^{n} \|U_m^T X_m - U_{m'}^T X_{m'}\|_F^2 \quad \text{s.t.} \quad U_m^T (X_m X_m^T) U_m = I_{d_m}. \tag{4}$$

Unfortunately this problem is well known to be NP-hard [25]. In order to overcome this issue, it is usual to introduce a unique common and low-dimensional representation $S \in \mathbb{R}^{d \times n}$. The problem is reduced to finding S and projections $\{U_m\}_{m=1}^M$ in order to maximize correlations between S and all the projected views $\{U_m^T X_m\}_{m=1}^M$. It leads to the formulation detailed in Eq. (5) with $\gamma = 0$. This relaxation of Eq. (4) is also solved using an eigenvalue decomposition. Indeed, the matrix S^* has columns equal to the k leading eigenvectors of the matrix $\sum_{m=1}^M X_m^T (X_m X_m^T) X_m$ [22].

Graph Multiview CCA (GMCCA). Chen et al. [23] have proposed GMCCA as an extension of MCCA in which graph-based prior knowledge on S, when available, can be incorporated. It can lead to an increase in clustering performance. The graph structure is taken into account by ensuring smoothness of S on the known graph, which is represented using the graph Laplacian matrix $L \in \mathbb{R}^{n \times n}$. By doing so, a graph-regularized CCA problem can be posed as follows:

$$\min_{(U_m)_{m=1}^M} \sum_{m=1}^{M} \|U_m^T X_m - S\|_F^2 + \gamma \mathrm{tr}(S L S^T) \quad \text{s.t.} \quad S S^T = I_d. \tag{5}$$

The solution S^* of this problem has columns equal to the k leading eigenvectors of the matrix $\sum_{m=1}^M X_m^T (X_m X_m^T) X_m - \gamma L$; see [22] for more details.

Due to the eigendecomposition involved in all these algebraic methods, they do not always scale well for large datasets (see Table 2). An alternative

method based on a variational approach applied to a probabilistic model has recently gained attention to reduce the computational cost. This relies on the work of Bach et al. [18], where it is shown that CCA has an equivalent probabilistic model.

2.2. Probabilistic CCA

PCCA. Bach et al.[18] have shown that linear CCA optimal projections, as in Eq. (3), can be obtained from a graphical model [18], where the views of an instance come from a latent variable (the common source) denoted z. Let us define a prior distribution on this latent space $p(z)$ and the conditional probability (also called decoders) for each view $p_{\theta_m}(x_m|z)$ which is the probability to have a certain view x_m given the latent vector z and parameters θ_m, hence[5,6] we have $\forall m \in \{1,2\}, \forall z \in \mathbb{R}^d, \forall x_m \in \mathbb{R}^{d_m}$:

$$\begin{aligned} z &\sim \mathcal{N}(0, I_d); \\ x_m &\sim p_{\theta_m}(x_m|z) = \mathcal{N}(W_m z + \mu_m, \Psi_m) \end{aligned} \tag{6}$$

with $\mu_m \in \mathbb{R}^{d_m}$, $W_m \in \mathbb{R}^{d_m \times d}$ and $\Psi_m \in \mathbb{R}^{d_m \times d_m} \succcurlyeq 0$ (positive semidefinite). We collect the trainable parameters in θ_m as $\theta_m = (W_m, \mu_m, \Psi_m)$. The optimal parameter θ^* is computed by maximizing the data log-likelihood with respect to $\theta = (\theta_1, \theta_2)$:

$$\log p_\theta(X_1, X_2) = \sum_{i=1}^{n} \sum_{m=1}^{2} \log \int_{\mathbb{R}^d} p_{\theta_m}(X_m^i|z) p(z) dz. \tag{7}$$

The parameter θ^* is the one for which the dataset $X = (X_1, X_2)$ is the most probable for p_θ, and therefore the best parameter to explain the data. Let us introduce the distribution p_m that is the unknown true distribution of data view m. Thanks to Bayes theorem, the optimal encoder distributions $p_{\theta_m^*}(z|X_m^i)$ are perfectly defined, and we have $\forall i \in [\![0, n]\!]$:

$$p_{\theta_m^*}(z|X_m^i) = \frac{p_{\theta_m^*}(X_m^i|z)p(z)}{p_m(X_m^i)}. \tag{8}$$

[5]Notation $z \sim p$ means z follows distribution p.

[6]Abuse of notation: any distribution p is indiscriminately written as $p(x)$ or p.

The expectation of optimal decoder is then exactly the optimal projection (cf. Eq. (3)) coming from CCA (cf. Eq. (1)):

$$\mathbb{E}_{z\sim p} p_{\theta_m^*}(z|X_m^i) = M_m^T {U_m^*}^T X_m^i \tag{9}$$

The solution is known up to some arbitrary matrices $M_m \in \mathbb{R}^{d\times d}$ such that $M_1^T M_2 = P_d$ where P_d is a diagonal matrix of the first d canonical correlations [18]. In this framework, CCA has a natural multiview extension to $M > 2$. To do so we introduce as many decoders as the number of views in data. We will use such an extension, while incorporating graph regularization like in [23].

Yet, solving the inference problem for a model such as Eq. (6) (i.e., a multi-dimensional probability distribution) is often intractable: first because it requires maximization of the log-likelihood and thus to integrate over all the latent spaces, and, second, because the true distribution p_m of each view is not known. Even if we were able to compute θ^*, we could not compute the decoder distributions (Eq. (8)). A variational approach solves these issues, as we will see next in recalling the method of variational autoencoder [26].

With that, probabilistic CCA solves the problem in $\mathrm{O}(n)$ and is thus scalable. Moreover it opens up to perform inference tasks such as generating and recovering missing views. Conversely, GMCCA has initially the advantage of adding a prior information (coming as a graph) over data structure, so as to compute better low dimensional representation. This prior imposes a smoothness property on this representation such that the common view S is smooth on the associated graph of Laplacian L. While it can be seen as a prior, it acts in the problem as an additional regularization term and the solution comes with the additional cost of requiring an eigendecomposition. Hence the method incurs $\mathrm{O}(n^2)$; therefore it does not scale well. Our objective is to get the best properties of both model, by forming a fully probabilistic CCA model while having such prior on the graph.

2.3. Variational bound and graph autoencoder

VAE. Kingma et al. [26] have shown that by introducing parametric distributions $q_\eta(z|X^i) = q_\eta(z|X_1^i, X_2^i)$, with parameters η, instead of the intractable distribution $p_\theta(z|X^i) = p_\theta(z|X_1^i, X_2^i)$, one can lower bound the log-likelihood in Eq. (7). This lower bound is referred to as the **evidence lower bound objective** (ELBO), given as:

$$\log p_\theta(X_1, X_2) \geq \sum_{i=1}^{n} \mathbb{E}_{z\sim q_\eta(z|X_1^i, X_2^i)}[\log(p_\theta(X_1^i, X_2^i|z))] - D_{KL}(q_\eta(z|X_1^i, X_2^i)\|p(z)). \tag{10}$$

The first term of ELBO ensures a correct data reconstruction due to the encoded latent representations. The second term acts as a regularizer ensuring that the posteriors distributions q_η for each multiview instance remain coherent in the latent space. It corresponds to the loss function of the variational autoencoders, used in most existing variational CCA methods.

ELBO is easier to approximate than the data log-likelihood so we maximise this lower bound with respect to both θ and η. Moreover this formulation gives directly the decoders as q_{η^*} without requiring the knowledge of the true view distributions p_m.

Graph VAE (GVAE). It is possible to account for geometric structure by using the variational autoencoder extension proposed by Kipf et al. [19] for link prediction on graphs. In their single-view framework ($M = 1$), data reside on the nodes of a graph having a weighted adjacency matrix $A \in [0, 1]^{n\times n}$. On the contrary of Eq. (10) where the ELBO is a sum of terms depending only on one instance i of the data, here we have to introduce a latent matrix $Z \in \mathbb{R}^{d\times n}$ to express the probability to have graph A between n latent variables[7].

Then the graph-aware ELBO loss function is defined as :

$$\mathcal{L}_{ELBO} = \mathbb{E}_{Z\sim q_\eta(Z|X,A)}[\log(p_\theta(A|Z))] - D_{KL}(q_\eta(Z|X, A)\|p(Z)), \tag{11}$$

where $q_\eta(Z|X, A)$ is the parametric probability distribution for encoder which is now parametrized by a graph neural network, $p(A|Z)$ is the graph decoder distribution and $p(Z) = \Pi_{i=1}^{n} p(Z(:, i))$ is the prior on the latent space, taken as a multivariate normal distribution. Following [19], ELBO first term will ensure graph reconstruction from latent space, but it does not allow for data reconstruction (which is useless for link prediction task). On the contrary, the second term acts as a regularizer of the latent space. We

[7]Note that we could introduce the model with two latent variables only, and that would be enough to write the model for link prediction.

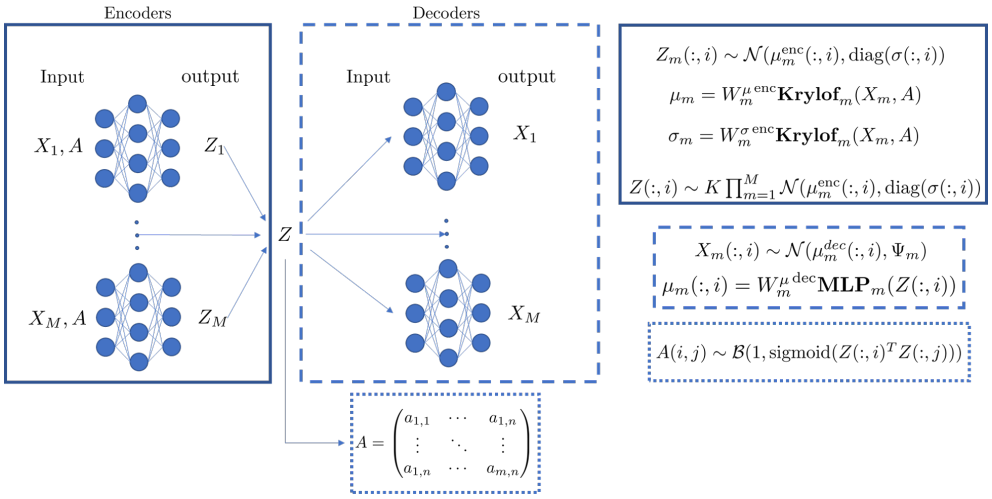


Figure 2: **Representation of MVGCCA.** All the views are encoded to their own latent space Z_m using the common graph. They are merged to form a common view Z. Finally, Z is tailored to decode all the views and original graph.

will use a similar approach to develop our method, while extending it for multiple views.

3. Variational graph MCCA

3.1. Model

We now present our contribution which consists of proposing a probabilistic multiview CCA model that can deal with missing views. Specifically, starting from Eq. (6), our contributions are as follows:

1. To extend the framework of [18] for $M > 2$. We introduce M decoders corresponding to the number of views. These decoders are parametrized by multilayer perceptrons (noted as **MLP**);
2. To take into account a prior graph structure on the latent space of z; inspired by [19], we add graph decoders into the PCCA frameworks;
3. To build a model that can deal with missing views in instances of datasets.

To this end, we define the following probabilistic model that forms the basis of our contribution, $\forall m \in \{1, \ldots, M\}, \forall z \in \mathbb{R}^d, \forall x_m \in \mathbb{R}^{d_m}$:

$$\begin{aligned} z &\sim \mathcal{N}(0, I_d); \\ x_m &\sim p_{\theta_m}(x_m|z) = \mathcal{N}(\boldsymbol{W}_m^{\mu\,\text{dec}}\mathbf{MLP}_m(z), \boldsymbol{\Psi_m}). \end{aligned} \tag{12}$$

This model relies on two hypotheses, in accordance to previous works [18, 19, 26]. First, z follows a multivariate normal distribution; this assumption may look unreasonable for most data, but is often used in variational autoencoders in the literature [26, 19] as this distribution of the prior is supposed to be mostly informative through its mean and variance, yet the specific shape is uninformative. In practice, it leads to good performance and it leads to a more convenient mathematical framework. In particular, it is easy to sample elements from Gaussian distribution and it leads to explicit formulation in the ELBO. The second hypothesis is that the conditional probability distributions of the view decoders are taken as multivariate Gaussian distributions parametrized by a multilayer perceptron for the mean, and a weight for the covariance matrix. Precisely, this hypothesis combined with the first one, allows to have an explicit form of Kullback-Leibler divergence on ELBO (Eq. (24)).

Next, we take into account a prior graph structure on the latent space of z using techniques inspired by [19]. Specifically we introduce a graph conditional probability distribution (graph decoder) which gives the probability to have an adjacency matrix $A \in [0,1]^{n\times n}$ given the n latent space vectors concatenated in the matrix $Z \in \mathbb{R}^{d\times n}$:

$$\begin{aligned} Z(:,i) &\sim \mathcal{N}(0, I_d); \\ A &\sim p_g(A|Z). \end{aligned} \tag{13}$$

In this context, we assume that all links of the graph are independent. Hence we introduce a weight decoder distribution $p_l(a = 1|z, z') \sim \mathcal{B}(1, \ell(z^T z'))$, parametrized by a Bernoulli law $\mathcal{B}$, where $\ell(\cdot)$ is the logistic sigmoid function and $a \in [0, 1]$ is the possible weight of a link. Thus the probability to have a graph defined by an adjacency matrix A given Z is:

$$p_g(A|Z) = \prod_{i=1}^{n}\prod_{j=1}^{n} p_l(A_{i,j}|Z(:,i), Z(:,j)). \tag{14}$$

3.2. Evidence lower bound objective

Using the hypothesis that all the views are independent one from others, as well as from the links, the log-likelihood can be written in the following form:

$$\log p_\theta(X, A) = \log\left(p_\theta(\{X_m\}_{m=1}^M)\, p_g(A)\right). \tag{15}$$

In this equation, $p_\theta(\{X_m\}_{m=1}^M)$ is the joint probability on all the views, X_1 to X_M; hence, this equation gives the probability of obtaining the multi-view dataset $X = \{X_m\}_{m=1}^M$ with the graph A given the model p_θ. We can explicitly express these probabilities as in Eq. (7):

$$\begin{aligned} \log p_\theta(\{X_m\}_{m=1}^M) &= \sum_{i=1}^{n}\sum_{m=1}^{M} \log \int_{\mathbb{R}^d} p_{\theta_m}(X_m^i|z)p(z)dz; \\ \log p_g(A) &= \sum_{i=1}^{n}\sum_{j=1}^{n} \log \int_{\mathbb{R}^d} p_l(A_{i,j}|z,z^{'})p(z)p(z^{'})dzdz^{'}. \end{aligned} \tag{16}$$

Since this computation is also intractable, we follow the methodology of Kingma et al. [26] in order to define an efficient approach. We introduce a parametric distribution written as in Eq. (17). As we are going to parametrize this distribution with graph neural networks, we need the notation $\mathcal{V}^l$ introduced earlier to refer to the features in the neighborhood of a node up to a distance l in the graph, i.e, we have:

$$q_\eta(Z|X,A) = \prod_{i=1}^{n} q_\eta(Z(:,i)|\mathcal{V}^l(X^i),A) = \prod_{i=1}^{n}\prod_{m=1}^{M} q_{\eta_m}(Z(:,i)|\mathcal{V}^l(X_m^i),A). \tag{17}$$

Were η_m is the trainable parameter for view m distribution encoder q_{η_m} and we have $\eta = (\eta_1, \ldots, \eta_M)$. The parameter l depends on the parametrization of the graph neural networks, which will be decided on the experimental part. Let us consider the Kullback-Leibler divergence between the parametric decoders and the decoders as:

$$D_{KL}(q_\eta(Z|X,A)\|p_\theta(Z|X,A)) \geq 0. \tag{18}$$

More explicitly, we have:

$$\int_{\mathbb{R}^{n\times d}} q_\eta(Z|X,A)\log\frac{q_\eta(Z|X,A)}{p_\theta(Z|X,A)}dZ \geq 0. \tag{19}$$

Using Bayes theorem, we can write Eq. (19) as:

$$\int_{\mathbb{R}^{n\times d}} q_\eta(Z|X,A)\log\frac{q_\eta(Z|X,A)p_\theta(X,A)}{p(Z)p_\theta(X,A|Z)}dZ \geq 0. \tag{20}$$

which develops into:

$$\begin{aligned}&\int_{\mathbb{R}^{n\times d}} q_\eta(Z|X,A)\log\frac{q_\eta(Z|X,A)}{p(Z)}dZ\\ -&\int_{\mathbb{R}^{n\times d}} q_\eta(Z|X,A)\log p_\theta(X,A|Z)dZ \geq -\log p_\theta(X,A).\end{aligned} \tag{21}$$

In other word, we have the inequality:

$$\log p_\theta(X,A) \geq \mathbb{E}_{Z\sim q_\eta(Z|X,A)}\log p(X,A|Z) - D_{KL}(q_\eta(Z|X,A)\|p(Z)). \tag{22}$$

The ELBO (lower bound of Eq. (22)) takes finally the following explicit form (similarly as in Eq. (10)):

$$\begin{aligned}\mathcal{L}_{ELBO} = &\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{E}_{\substack{z\sim q_\eta(z|\mathcal{V}^l(X^i),A)\\ z'\sim q_\eta(z'|\mathcal{V}^l(X^j),A)}}\log p_g(A_{i,j}|z,z')\\ &+\sum_{i=1}^{n}\sum_{m=1}^{M}\mathbb{E}_{z\sim q_\eta(z|\mathcal{V}^l(X^i),A)}\log p_{\theta_m}(X^i_m|z)\\ &-\sum_{i=1}^{n} D_{KL}(q_\eta(z|\mathcal{V}^l(X^i),A)\|p(z)).\end{aligned} \tag{23}$$

In this formula, we obtain a graph reconstruction term, an explicit data reconstruction term, and a regularizer in the latent space. All these terms involve the parametric view decoders q_η that we define next.

3.3. Parametric decoders

We choose in this model to parametrize each q_{η_m} by a multivariate Gaussian distribution with a **Krylov** Graph convolutional neural networks[8] (GCN) [27] as:

[8] log in $\log\sigma_m$ is applied element-wise on matrix σ_m.

$$
\begin{aligned}
q_{\eta_m}(z|\mathcal{V}^l(X_m^i), A) &= \mathcal{N}(\mu_m^{\text{enc}}(:,i), \text{diag}(\sigma_m^{\text{enc}}(:,i)).\\
\mu_m^{\text{enc}} &= \boldsymbol{W}_m^{\mu\ \text{enc}}\mathbf{Krylov}_m(X_m, A).\\
\log\sigma_m^{\text{enc}} &= \boldsymbol{W}_m^{\sigma\ \text{enc}}\mathbf{Krylov}_m(X_m, A).
\end{aligned}
\tag{24}
$$

where μ_m^{enc} and $\log\sigma_m^{\text{enc}}$ are matrix output by Krylov layer whose i-th column is, respectively, the mean and the diagonal of the covariance matrix of q_{η_m}. We use the log to ensure positivity of sigma.

The reason for the choice of a GCN is that they are efficient to extract feature information of a node considering its neighborhood [28, 29]. They have been widely used in node classification, node clustering, and other graph analytic tasks. Currently, many GCN have similar performance [30], and since the choice is not critical, we simply use a truncated Krylov GCN architecture [27] which has been proven to have good properties when stacking across graph layers.

3.4. Parameters of the model and inference

Globally, the trainable parameters (i.e., the ones to infer) of the models are the weights $\boldsymbol{\Psi_m}$, $\boldsymbol{W}_m^{\mu\ \text{dec}}$, $\boldsymbol{W}_m^{\sigma\ \text{enc}}$, $\boldsymbol{W}_m^{\mu\ \text{enc}}$, respectively the parameters of the multilayer perceptron $\mathbf{MLP}_m$ and of the Krylov GCN layers $\mathbf{Krylov}_m$.

In order to infer these parameters efficiently, we will use the ELBO of Eq. (23). Even if this ELBO is not separable in a sum of terms depending only of one instance i, it can be decomposed in terms containing only subsets of some instances i. This leads to a training strategy of the model in a batch manner, using a suitable optimization method. For every instance i, each view m (i.e., X_m^i) has it's own latent representation, which is computed similar to Eq. (9) i.e $\mathbb{E}_{z\sim q_{\eta_m}(z|\mathcal{V}^l(X_m^i),A)} = \mu_m^{\text{enc}}(:,i)$. These latent representations of views allow us to build the common latent representation of instance i as (see Fig. (2) for illustration):

$$
\mathbb{E}_{z\sim q_\eta(z|\mathcal{V}^l(X^i),A)} = \left[\sum_{m=1}^{M} \frac{\mu_m^{\text{enc}}(k,i)}{\sigma_m^{\text{enc}2}(k,i)} \Big/ \sum_{m=1}^{M} \frac{1}{\sigma_m^{\text{enc}2}(k,i)}\right]_{k=1}^{d}. \tag{25}
$$

These representations are used for the experiments presented later on.

3.5. Robustness and "views dropout"

The formula (25) comes from the choice of q_η as a product of multivariate Gaussian distribution with diagonal covariance matrix. This particular choice gives the model robustness property to deal with missing views. Indeed by model assumptions, each view of an instance i of the dataset is derived from a latent variable we seek for. For each view, the distribution $q_{\eta_m}(z|\mathcal{V}^l(X_m^i), A)$ gives the probability that the latent variable z generates the corresponding views. Thus we have $q_\eta(z|\mathcal{V}^l(X^i), A) = \Pi_{m=1}^M q_{\eta_m}(z|\mathcal{V}^l(X_m^i), A)$ which is the probability that z generates (all the views of) an instance i; hence the probability of an instance is given by the product of probability mass of the different views of this instance i. Thus, the more probability mass the different views m gives to z, the more likely z are the sources of i (i.e of these views). But these views do not contribute equally to the probability. Some views or groups of views contains more information than others about z, as can be seen in Eq. (25) by the fact that z is the barycenter (using the precision (inverse variance) as weight) of view's most probable latent space vector (i.e the mean of decoders).

In the case in which some views are missing, if the sum of precisions associated to missing views is not large compared to the one associated to existing views, the computation of Eq. (25) without these views will be still an accurate approximation of z. So given enough views, we could still compute a good approximation of the common latent representation. This potential ability to deal with missing views is successfully confirmed in the experiments reported under Section 5.3.2 and 5.3.3. There is no such direct possibility with other variational CCA approaches. Indeed in the 2-views variational model VCCA [20], q_η is parametrized only by one of the two views, hence this cannot be robust to the absence of the corresponding view for some instances. The improved model VCCA-private in the same article also suffers from this. The more recent variational model VPCCA [21] can deal with limited numbers of views available, but needs to have these views available for each instance to compute the low dimensional common latent space which makes it obviously not robust to real missing views situations where different instances could have different missing views (see Fig. (3)). The present model exploits the maximum amount of information available for each instance.

In order to reinforce the robustness of the proposed model, we introduce for the experiments the novel notion of "views dropout". This consists of randomly ignoring certain views during the encoding part of the training phase while still asking the model to be able regenerate all the views as well

as the adjacency matrix. This procedure is more detailed and evaluated in experiment part 5.3.2 and 5.3.3.

As a summary of this methodological part, we repeat that our model is the only CCA model which is scalable, graph aware, and robust to missing views.

Remark: The choice for all q_{η_m} involves that q_η (cf. Eq. (17)) is not a correctly normalized distribution. However, q_η is proportional to a distribution (there is a constant K such that Kq_η is a distribution). If we resume the calculation started in Eq. (18) with Kq_η it will lead to the same $ELBO$ up to an additive and a multiplicative constant that does not change inference, so we can work as if q_η was a distribution.

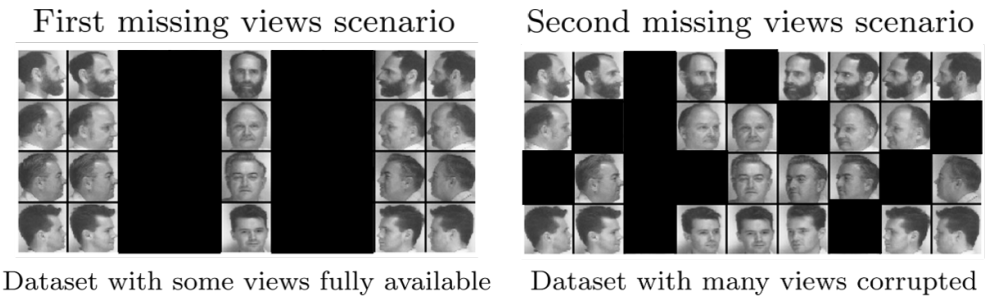


Figure 3: **Multiview dataset with missing views.** On the left, one can see a multivew dataset where some views are missings but some are fully available. Variational model such as VCCA (limited to two-view dataset) and VPCCA can deal with these kind of data. We evaluate MVGCCA in this scenario in Section 5.3.2. On the right one can see a dataset were many views are corrupted. This dataset is a much more realistic representation of errors which can be found on real dataset. The proposed model can deal with these kinds of missing data during both training and testing. MVGCCA exploits the maximum amount of information available for each instance. MVGCCA is evaluated in this scenario in Section 5.3.3.

4. Datasets

4.1. UCI handwritten digits dataset

UCI Handwritten Digits Dataset[9] is a multiview dataset of $n = 2000$ samples images representing digits. Each image has a label from 0 to 9 (200 instances for each), and 6 views with different dimensions: $d_1 = 76$,

[9] `archive.ics.uci.edu/ml/datasets/Multiple+Features`

Dataset	**uci7**			**uci10**			**Recommendation**		
Metric	Acc.	ARI	ARI2	Acc.	ARI	ARI2	Prec.	Recall	Mrr
PCA	0.84	0.55	-	0.69	0.42	-	0.1511	0.0795	0.3450
GPCA	0.93	0.71	0.77	0.87	0.63	0.62	0.1578	0.0831	0.3649
MCCA	0.86	0.66	-	0.76	0.59	-	0.0815	0.0429	0.2225
GMCCA	**0.95**	**0.83**	0.84	0.90	0.69	0.71	**0.2290**	**0.1206**	**0.4471**
MVGCCA	**0.95**	0.82	**0.85**	**0.94**	**0.74**	**0.77**	0.1753	0.0583	0.4432
Dataset							**Recommendation Large**		
Metric							Prec.	Recall	Mrr
MVGCCA							0.1745	0.0960	0.4301

Table 2: Results of experiments on the different datasets and tasks; see text for the detailed discussion. Acc. stands for accuracy in classification; ARI for adjusted rank index in clustering tasks: ARI1 if using K-means and ARI2 if using spectral clustering. For the Recommendation task, Prec. is precision and Mrr is the mean reciprocal rank.

$d_2 = 216$, $d_3 = 64$, $d_4 = 240$, $d_5 = 47$ and $d_6 = 6$. These views correspond to specific transformations of the original image: Fourier coefficients of the character shapes $X_1 \in \mathbb{R}^{d_1 \times n}$; profile correlations $X_2 \in \mathbb{R}^{d_2 \times n}$; Karhunen-Loeve coefficients $X_3 \in \mathbb{R}^{d_3 \times n}$; 240 pixel averages in 2 x 3 windows $X_4 \in \mathbb{R}^{d_4 \times n}$; Zernike moments $X_5 \in \mathbb{R}^{d_5 \times n}$; and 6 morphological features $X_6 \in \mathbb{R}^{d_6 \times n}$. Clustering, classification and reconstruction tasks are performed on this whole dataset (uci10); also, we consider as in [23] for comparison, a partial version (uci7) where classes 0, 5 and 6 have been removed. The obtention of a prior graph is an important step for the analysis. As the objective is to be able to compare the method to existing one, like [23], the choice is made to follow exactly the method described in Chen et al. [23] (Section VII.C) to build the prior graph over data.

4.2. Twitter friend recommendation

A multiview dataset[10] based on post from Twitter has been proposed in [6]. It consists of multiview representations of messages of users. They took 1% of the publicly available users data in April 2015. They removed all tweets that are not in english, and those from users who did not post between January and February 2015[11]. Finally they only kept the last 100 tweets from the remaining users, yielding $n = 102327$ users. These data for each user

[10] `http://www.cs.jhu.edu/~mdredze/data/`

[11] There are other minor exclusion criteria which may be found in [6].

Figure 4: **Illustration of two-view MNIST dataset.** Three samples of two-view MNIST dataset. For each digit (1, 5 and 8) we display on the left : original image, middle: first (rotated) view, right: second (noisy) view.

are in the form of 6 1000-dimensional views: EgoTweets, MentionTweets, FriendTweets, FollowersTweets, FriendNetwork, and FollowerNetwork. A task of friend recommendation is performed as follows. The followed accounts are known for each user; given a highly followed account and a part of their followers, the goal is to determine, for each other users, whether or not he will follow this account after March 2015. For this task, a graph based on the Twitter dataset is built as in [23] with the views Egoweets, FollowersTweets, and FriendNetwork.

4.3. Two-view MNIST noisy dataset

Two-view MNIST noisy dataset has been proposed by Wang et al. [15]. It is a two-view dataset built from the famous MNIST handwritten digits 28×28 dataset. MNIST dataset is composed of a training set with 50000 instances, a validation set with 10000 instances and a test set with 10000 instances. We merge them. The first view is obtained after performing a rotation to the 70000 images. Angles of each rotation have been sampled from continuous uniform distribution $\mathcal{U}([-\pi/4, \pi/4])$ with $-\pi/4$ as minimum value and $\pi/4$ as maximum value. For the second view, we choose randomly another image with identical labels from MNIST and we add a noise sampled from uniform distribution $\mathcal{U}([0, 1])$ to each pixel. Finally, we truncate pixels values to keep them between 0 and 1. This data has no pre-defined graph structure. We build the prior graph by connecting each instance with a probability 1/10 (resp. 1/1000) to another instances with the same labels (resp. different labels). At the end each instance is connected to nearly 10% of instance with a different label. Hence this graph structure gives some (noisy) supplementary information about relation between instances of this dataset. Finally, to evaluate our model we split the dataset in a training set, a validation set and a test set of the same size as at the beginning. A visualisation of this dataset can be seen on Figure 4.

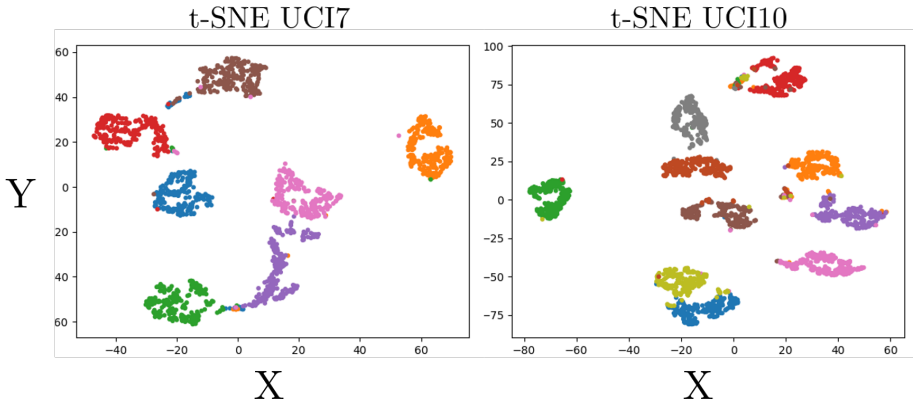


Figure 5: t-SNE visualisation in 2D of the latent space ($d = 3$) for the uci7 and uci10 dataset. Each color represents a different class.

5. Experiments

All architectures and hyperparameters specified here are the same for all experiments[12], unless otherwise indicated in the text. We pre-process all the views: each view is centered and normalized by standard deviation. For each dataset, the graph adjacency matrix is rescaled with its maximal entry and diagonal coefficients are set to 1.

Decoders: Mean decoders computed by $\mathbf{MLP}_m$ for each view m are used with a ReLU non linearity except the last layer which is a linear transformation. Covariance matrix decoders $\mathbf{\Psi}_m$ are trained as a scaled identity matrix: $\mathbf{\Psi}_m = ((\sigma_m + 10^{-6})^2) I_{d_m}$. This choice reduces the complexity and improves robustness, and we have seen in experiments that it does not decrease performance.

Encoders: Krylov GCN layers [27] encoding the mean and variance in q_η, have a depth of $l = 4$ hop neighborhood.

General features: Batch size is set to 512. A dropout regularization of rate 0.5 is also applied after all the hidden layers. The Adam optimizer is used for training. A decay learning rate is applied: $l_r 1.1^{-50\frac{e}{E}}$, where e is the current epoch and E the maximal number of that is set to 600 epochs by default.

[12] Code available : `https://github.com/Yacnnn/MVGCCA`

5.1. Classification and clustering

We performed classification and clustering experiments. A first experiment involves a comparison to GMCCA (which is the only method which can deal with graph). And a second one shows comparisons to many other non linear method such as Deep CCA, Kernel CCA, Variational CCA, etc (cf Tab. (3)).

5.1.1. UCI

The model was trained on uci7 and uci10 datasets. The loss function was the ELBO from Eq. (23), trained with a batch of dataset. For each batch, the graph used is the subgraph of samples in the batch. The latent space is of dimension $d = 3$. We perform a grid search over the learning rate (1e-3,1e-4), number of hidden layers (3,4) and hidden layers size (512, 1024). We also decide to apply a decay learning rate or not, and to look for decoders covariance matrix as full matrix or scalar matrix based on a grid search. For each combination of these parameters we trained the algorithm three times, on 600 epochs and we save each latent embedding every 100 epochs (to perform early stopping).

We take 90% instance of dataset as the train set and the remaining 10% as test set. We perform a 5-fold cross validation with SVM-RBF accuracy to find optimal parameters (MVGCCA and SVM-RBF hyperparameters and early stopping step) on the train set. Finally, for the best parameters, we train the SVM-RBF on the train set and evaluate it on test set. We also perform a K-means and spectral clustering on the whole embeddings (the train and test sets). A 2D t-SNE projection of latent space can be visualized in Figure 2.

Because of the small size of UCI dataset, the ouput of this procedure depends on initial choice of dataset splitting; so in order to overcome this issue, we perform this experiment 100 times and average the results. Following [23], results are compared to PCA (applied on concatenated views), graph-regularized PCA, MCCA, and GMCCA. In order to make fair comparison in terms of hyparameter tuning, all the experiments were done with the same protocol for all the methods.

The results are summarized in Table 2. We see that MVGCCA is competitive in both classification (Acc.) and clustering tasks (ARI if using K-means & ARI2 if using spectral clustering). It achieves the best performance on the more complex dataset uci10 over other graph aware methods. This indicates that the graph structure is well encoded in the latent space.

Remark: In using SVM-RBF, we have conducted a grid search (with sklearn package) for the following parameters: $\mathbf{C}^{13}$: 7 values logarithmically spaced in [10^{-3},10^{3}] and **gamma**: 7 values logarithmically spaced in [10^{-3},10^{3}]. The K-means and the spectral clustering (with **gamma** = 5) are performed with the default parameters of sklearn.

5.1.2. Two-view MNIST

This dataset is composed of a training, validation and test sets. For this experiment we train our model only on the training set. We performed a small grid search based on insight given by previous experiments; hence we only tune the latent space dimension $d = \{30, 60\}$ and if we use a scalar decoder or not. We used a 4 hidden layers and the hidden layers size is set to 1024 for both encoders and decoders. We trained the algorithm during 200 epochs and we save each latent embedding every 10 epochs after 50 epochs to perform early stopping. Once the model has been trained, the latent space of train, validation and test set is then inferred from the model. In order to perform fair comparison with others variational method VCCA [20] and VPCCA [20], we suppose that only the first view is available to produce the test set latent space (first scenario of Figure 3), it is equivalent to use Z_1 as latent representation. We use theses representations for classification. We use VPCCA [21] and [15] as the baseline. We train a Linear SVM with regularization parameter $\mathbf{C}$ having possibly 8 values that are logarithmically spaced in [10^{-3},10^{3}]. We used the validation set to select best parameter for this linear SVM, the dimension of our latent space, and to decide to stop early the epochs. Finally, we trained the linear SVM on both training and validation set with these parameters and evaluate our method on the test set. On contrary to the UCI dataset, two-view MNIST is a large dataset, so the results of these experiments have been averaged only over three runs. The results of these experiments are summarized in Table 3.

As it can be seen, the results are competitive against the best methods of the state of art, thanks to the information brought by the graph structure. This illustrates again how CCA can benefit from taking into account some geometrical structure in the multivew dataset.

Dataset	**Two-view MNIST**
Metric	Acc.
Linear CCA	0.804
SpliAE	0.881
Kernel CCA	0.949
Deep CCA	0.971
DCCAE	0.978
VCCA	0.970
VCCA-(p)	0.976
VPCCA	0.981
MVGCA	**0.985**

Table 3: **Results of classification with a linear SVM on two-view MNIST**; see text for the detailed discussion. The baseline is issue from Wang et.al [15] and Karami et al.[21]. The baseline involve linear CCA [8]; non linear method: Kernel CCA [17], Deep CCA [13], Deep CCA autoencoders [15], and probabilistic model Variational CCA (-private) [20] or Variational Probabilistic model [21]. For VCCA, VPCCA and MVGCCA, all views are available during training time, while only the first view is available for inference. MVGCCA and VPCCA are the only model which can deal with more than two views.

5.2. Twitter friend recommendation

For this dataset and the recommendation task, no further hyperparameter tuning is done and values from previous experiments are used with a learning rate $= 10^{-4}$, number of hidden layers $= 4$, hidden layers size $= 1024$, number of epochs $= 600$, a decay learning rate and a scalar covariance matrix for decoders. The parameters of the methods used for comparison are extracted from [16, 23] with their best parameters. The twitter dataset is large with more than 100,000 users, which makes it intractable for existing methods. Hence, 2506 twitter users are randomly selected from the database as in [23] for fair comparison.

The 20 most followed accounts (over the whole dataset) are selected, For each of these, 10 users following them are chosen (at random) and the average representation from latent space is computed. These average profiles will represent the typical users who follow the 20 most followed accounts. The latent space is set to dimension $d = 5$. Finally, the cosine similarity is computed between this average profile and the one of the $L = 100$ closest users to these representations. If one of these 100 users actually followed the

[13]Regularizer parameters of Linear SVM. See sklearn implementation of Linear SVM.

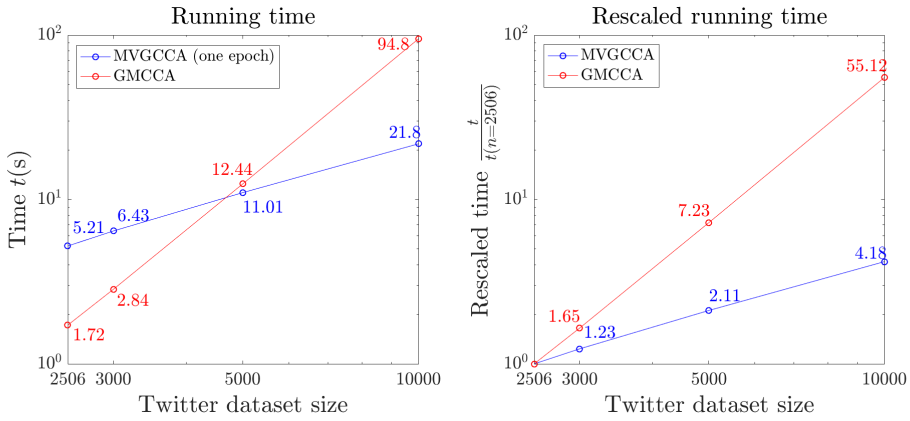


Figure 6: Log-log plot of running time between the two graph CCA methods: MVGCCA and GMCCA. We measure the running time for a run of GMCCA and one epoch of MVGCCA for four twitter dataset size ($n = 2506, 3000, 5000, 10000$). On the right figure we rescaled the running time by the running time at $n = 2506$. One can see that the GMCCA running time grows much more quickly than for MVGCCA.

initially chose account, this is considered as a good friend's recommendation. To assess performance, precision, recall, and Mrr (mean reciprocal rank) metrics are computed (averaged for the 20 most followed accounts). This experiment has been repeated and averaged over 100 sampling of 2506 users. The results are in Table 2 (right).

The performance of MVGCCA is comparable (except for recall) to that of GMCCA, which is currently the best method for this task. We recall here that the results are assessed on a limited dataset that all methods can process while our model could process a larger sample of dataset. To check that, we re-run the same experiment on larger dataset in order to show the scaling of the running time for a dataset from $n = 2506$ to $n = 10000$. The result is displayed in Fig. 6. We observe that the scaling is, as expected, better for MVGCCA and, for data of size $n = 5000$ of larger, MVGCCA epochs are quicker despite the overhead in computing time due to the use of GCN. Considering that after 100 epochs we already have goods results.

In order to give insight of what happens when we deal with a larger dataset, we also report the result of the recommendation task for the largest scale experiment, i.e with dataset 5 times larger. Here, we consider $n = 12530$ users and the 100 most followed accounts. We choose 50 users following these most followed accounts to compute average representation. The same

training parameters are used but latent space dimension which is set to $d = 10$. For evaluation, we take $L = 500$ in order to keep the same difficulty (we have 5 times the number of users) so as to make this experiment comparable to the previous one. A graph based on the Twitter dataset is built as before, from the exact same views Egoweets, FollowersTweets, and FriendNetwork, but with the neighbor parameters equal to 100 (instead of 50 before). Please refer to [23] to have more information about this parameter. The experiments have been averaged over 20 sampling of 12530 users. The result of this larger scale experiment is in Table 2 (last line). As it can be seen, the performance is quite the same on precision and Mrr metrics while there is an improvement for the recall metrics. It means that, thanks to the larger amount of data, the algorithm is capable of finding more of what it is possible to be find, while conserving a good precision. This results in a greater ability to perform recommendation.

Remark: We did not process a larger experiment because of the limitations associated to graph building procedure. On a data coming naturally with a graph structure, this problem would not exist; we choose here this dataset for the sake of comparison on a situation studied in the literature.

5.3. Inference

In a final experiment, we illustrate the robustness of the model to missing views for some instances, and its ability to reconstruct these missing views. This property of the model is unique among other graph aware CCA methods. To do so, we rely on the probabilistic nature of our model. Once the model has been trained, for a new instance not seen during training we apply the formula of Eq. (25) restricting the sum to the available views. Then this approximation of the latent representation z^{approx} is used to regenerate any missing views m as:

$$x_m^{regenerate} = \mathbb{E}_{x \sim p_{\theta_m}}(x|z^{approx}) = \boldsymbol{W}_m^{\mu\ \mathrm{dec}} \mathbf{MLP}_m(z^{approx}). \tag{26}$$

5.3.1. Recovering missing views

We consider the following scenario: We train the model on a training set (90% of data) where all the views are available for all instances. Then, for every $k \in [\![0, 4]\!]$, we randomly select 10 instances in a test set[14] for which we

[14] Whose elements have not been seen during training.

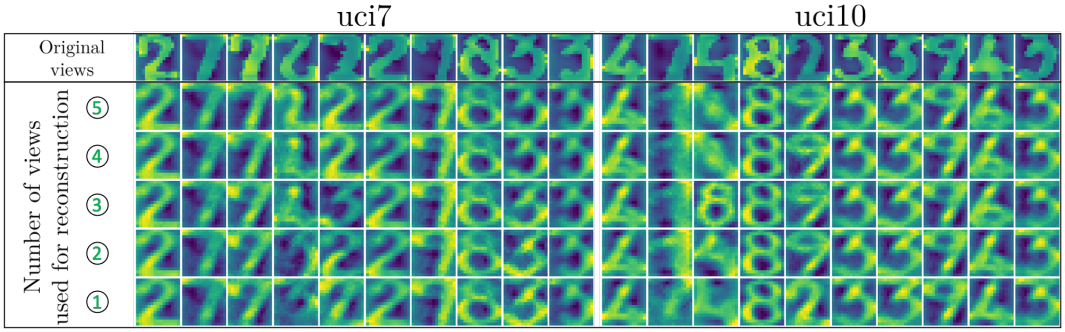


Figure 7: **UCI 4th view reconstruction.** We see here the result of the reconstruction experiment of the 4th view. The first line corresponds to the original view. Every figure in line $l > 2$ corresponds to a reconstruction from $7 - l$ views. We did not use views dropout in this experiment.

remove the 4th view and k additional views[15]. Note that, in this dataset, the 4th view is a subsampling of the original image. Using the other views, we regenerate 4th views according to Eq. (25) restricted to available views.

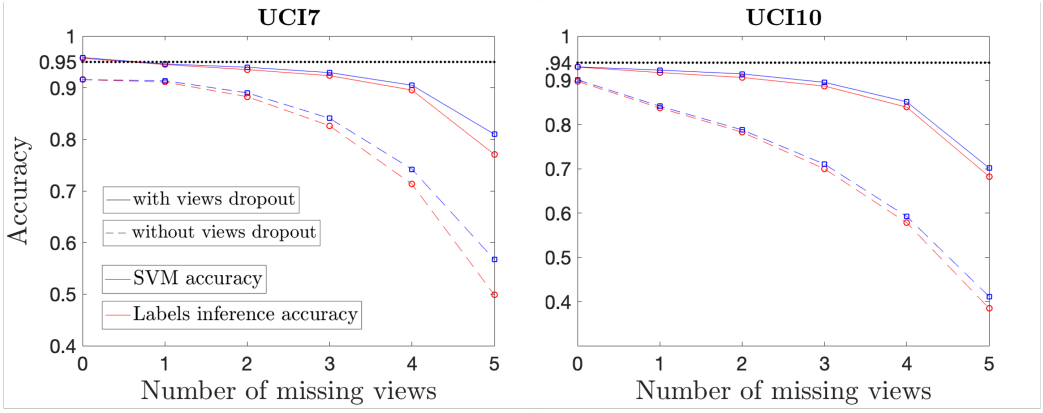
The results are given in Fig. 7. We see that in the case of uci7 the reconstruction goes very well in the majority of cases whatever the number of missing views. Still, there are some problematic cases. For instance, in column 4 we can clearly see that the original view is upside down, while in column 5 the original view is degraded – this shows that in both cases we have an atypical point of the dataset. Also, in column 7 we see some confusion between a 7 and a 9. The right part of the Figure is for uci10 which is a more difficult dataset than uci7. The method has additional difficulties to regenerate some views, because of some confusions between 4 and 6, and between 7 and 1. Still, this difference between uci7 and uci10 suggests that a training on a larger database should solve these problems. Anyway, we see that the majority of the proposed reconstructions from the model are very close to original views, thus demonstrating experimentally the ability of our model to reconstruct missing views.

5.3.2. Robustness wrt. missing views – first scenario

This scenario corresponds to the first of Figure 3. In this experiment the quality of the reconstruction is evaluated more quantitatively and this will

[15]The k views removed are the k views with the smallest dimension.

a. First scenario



b. Second scenario

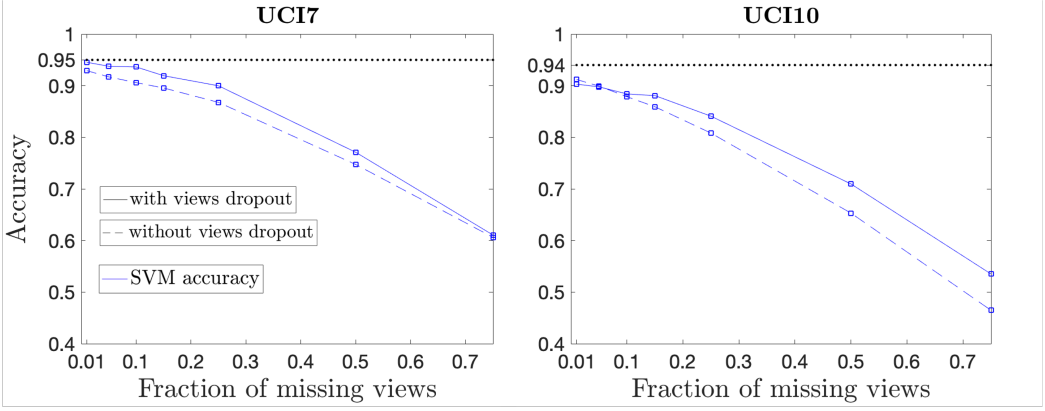


Figure 8: **Illustration of MVGCCA robustness**. See section 5.3.2 and 5.3.3 and Fig. 3 for details on both scenario. **a. First scenario:** The accuracy of clustering is compared for the two procedures of test (by SVM-RBF or inference of 1-hot encoding), without and with "views dropout". The black dotted line corresponds to accuracy found in Section 5.1.1 with all views. Note that we did not search for optimal parameters here, so it is expected to found a lower accuracy for 0 missing views. Still, for uci7, we obtain a better accuracy when using "views dropout". **b. Second scenario:** We evaluated on the SVM-RBF accuracy in this scenario. One can see similar behavior. The more the fraction of missing views, the more the performance decrease. The procedure of "views dropout" helps somehow the algorithm to deal with missing views, with a diminished the effect in comparison to the first scenario. Globally, this scenario seems to be harder, yet MVGCCA is currently the only method to deal with it.

show the robustness of the proposed model when only a subset of views are (fully) available. To do so, we work with regular UCI and an extended UCI where we add to the training set a 7th view that corresponds to a one hot encoding of the instance label. Then, we train MVGCCA on this two sets. Finally, we evaluate accuracy on test set in two manners:

- For regular UCI, we train a SVM-RBF on train set embeddings from trained model. Then we evaluate this SVM on test set embeddings where some views have been removed. These embedding are computed as in section 5.3.1.

- For extended UCI, we regenerate the 7th views of test set instances. This directly gives us an estimation of their labels. Once again, some views are removed.

For a given number of available views $v \in [\![1, 5]\!]$, we consider any possible combination of views to form novel datasets, and we average the obtained accuracies (e.g if $v = 2$ the set of available views considered are $\{(1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 3), \ldots, (5, 6)\}$). It allows us to obtain a result that does not depend on the nature of the missing views. Moreover, we repeat this experiment 10 times and average the results. They are provided in Figure 8.a as dashed line. As we can see, the model can deal with a small number of missing views; still the performance decreases with the number of these missing views and that is not surprising.

In the same figure, we show results with continuous line where we robustify the method by applying a "*views dropout*" during training phase. This consists in randomly removing between 1 and 5 views to each instance of the batch size for the encoding part of the model only. Hence the model is trained in order recover all views from only a subset of views. As we can see this procedure makes the model very robust: for instance, when 4 views are missing, the accuracy with *views dropout* remain better than using the model without *views dropout*, even with a smaller number of missing views. This allows even for better results to be obtained when no view is missing.

5.3.3. Robustness wrt. missing views – second scenario

This scenario corresponds to the second of Figure 3. In the previous experiment, limited subsets of available views were available for generating the latent space but all instances had the same set of available views. In real scenario different instances may have different missing views. We will

evaluate MVGCCA in this scenario. Once again we will train MVGCCA on train set (90% of the dataset) and then infer the test set latent space after having randomly removed some views. We removed a percentage $r = \{1\%, 5\%, 10\%, 15\%, 25\%, 50\%, 75\%\}$ of views on the test sets while asserting that each instance will have at least one views available. We repeat the experiment 10 times and for each we evaluated 10 times each percentage of missing views r with and without views dropout.

As can be seen in Figure 8.b, we have a similar behavior than for previous experiments. The method is able to deal with this type of corrupted dataset. This is the first method to do so in the literature. The procedure of "views dropout" is again useful somehow (this time by removing between 1 and 3 views to each instance), but its effect is diminished as compared to scenario 1.

Remark: As discussed before, in order to compute a low dimensional representation when some views are missing, we restrict the formula of Eq. (25) to available views. In this second scenario, we need to make some adjustement in order to compute $\mu_m^{\text{enc}}(:, i)$ and $\sigma_m^{\text{enc}}(:, i)$ from Eq. (25), as they depend on the views m of neigbours of i (see Eq. (24)); these views could be missing. We overcome this issue by replacing in $q_{\eta_m}(z|\mathcal{V}^l(X_m^i), A)$ the adjacency matrix A by A_m which is the adjacency matrix where unavailable views have been suppressed. Thus $\mathcal{V}^l(X_m^i)$ contains only available views m in the neighborhood of instance i.

6. Conclusion

We proposed MVGCCA, a novel multiview and non linear extension of CCA based on a Bayesian inference model. The proposed model is scalable, and can take into account the available graph structural information from the data. We have also proposed also a robustification method to handle missing data by applying "*views dropout*" during training. The probabilistic graphical nature of the model can be used for other tasks, such as addressing link prediction for multiview datasets, and that is a perspective for future work.

References

[1] M. E. Quemener, "SIDUS", the solution for extreme deduplication of an operating system, The Linux Journal (2014).

[2] Y. Kaloga, P. Borgnat, S. P. Chepuri, P. Abry, A. Habrard, Multiview variational graph autoencoders for canonical correlation analysis, in: 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021. arXiv:2010.16132.

[3] Y. Yamanishi, J.-P. Vert, A. Nakaya, M. Kanehisa, Extraction of correlated gene clusters from multiple genomic data by generalized kernel canonical correlation analysis, Bioinformatics 19 (2003) i323–i330.

[4] R. Arora, K. Livescu, Multi-view cca-based acoustic features for phonetic recognition across speakers and domains, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 7135–7139.

[5] R. T. Collins, A. J. Lipton, H. Fujiyoshi, T. Kanade, Algorithms for cooperative multisensor surveillance, Proceedings of the IEEE 89 (2001) 1456–1477.

[6] A. Benton, R. Arora, M. Dredze, Learning multiview embeddings of twitter users, in: Proc. Annual Meeting Assoc. Comput. Linguistics, volume 2, 2016.

[7] Y. Li, M. Yang, Z. Zhang, A survey of multi-view representation learning, IEEE Transactions on Knowledge and Data Engineering 31 (2019) 1863–1883.

[8] H. Hotelling, Relations between two sets of variates, Biometrika 28 (1936) 321–377.

[9] J. Kettenring, Canonical analysis of several sets of variables, Biometrika 58 (1971) 433–451.

[10] K. Chaudhuri, S. Kakade, K. Livescu, K. Sridharan, Multi-view clustering via canonical correlation analysis, in: Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, Association for Computing Machinery, 2009, p. 129–136.

[11] S. Akaho, A kernel method for canonical correlation analysis, in: In Proceedings of the International Meeting of the Psychometric Society (IMPS2001, Springer-Verlag, 2001.

[12] L. P. Ling, C. Fyfe, Kernel and nonlinear canonical correlation analysis., International Journal of Neural Systems 10 (2000) 365–377.

[13] G. Andrew, R. Arora, J. Bilmes, K. Livescu, Deep canonical correlation analysis, volume 28 of *Proceedings of Machine Learning Research*, PMLR, Atlanta, Georgia, USA, 2013, pp. 1247–1255.

[14] A. Benton, H. Khayrallah, B. Gujral, D. A. Reisinger, S. Zhang, R. Arora, Deep generalized canonical correlation analysis, in: Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), Florence, Italy, 2019, pp. 1–6.

[15] W. Wang, R. Arora, K. Livescu, J. Bilmes, On deep multi-view representation learning, volume 37 of *Proceedings of Machine Learning Research*, PMLR, Lille, France, 2015, pp. 1083–1092.

[16] X. Chang, T. Xiang, T. M. Hospedales, Scalable and effective deep CCA via soft decorrelation, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, 2018, pp. 1488–1497.

[17] D. Lopez-Paz, S. Sra, A. Smola, Z. Ghahramani, B. Schölkopf, Randomized nonlinear component analysis, 2014.

[18] F. R. Bach, M. I. Jordan, A probabilistic interpretation of canonical correlation analysis, Technical Report, Department of Statistics, University of California, Berkeley, 2005.

[19] T. N. Kipf, M. Welling, Variational graph auto-encoders, arXiv:stat.ML 1611.07308 (2016). `arXiv:1611.07308`.

[20] W. Wang, H. Lee, K. Livescu, Deep variational canonical correlation analysis, ArXiv abs/1610.03454 (2016).

[21] M. Karami, D. Schuurmans, Variational inference for deep probabilistic canonical correlation analysis, 2020. `arXiv:2003.04292`.

[22] J. Chen, G. Wang, Y. Shen, G. B. Giannakis, Canonical correlation analysis of datasets with a common source graph, IEEE Transactions on Signal Processing 66 (2018) 4398–4408.

[23] J. Chen, G. Wang, G. B. Giannakis, Graph multiview canonical correlation analysis, IEEE Transactions on Signal Processing 67 (2019) 2826–2838.

[24] T. D. Bie, B. D. Moor, K. Arenberg, On the regularization of canonical correlation analysis, 2002.

[25] J. Rupnik, P. Skraba, J. Shawe-Taylor, S. Guettes, A comparison of relaxations of multiset cannonical correlation analysis and applications, CoRR abs/1302.0974 (2013). arXiv:1302.0974.

[26] D. Kingma, M. Welling, Auto-encoding variational bayes, in: 2nd International Conference on Learning Representations, ICLR, 2014. URL: http://arxiv.org/abs/1312.6114.

[27] S. Luan, M. Zhao, X.-W. Chang, D. Precup, Break the ceiling: Stronger multi-scale deep graph convolutional networks, in: Advances in Neural Information Processing Systems 32, 2019, pp. 10945–10955.

[28] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Advances in Neural Information Processing Systems 29, 2016, pp. 3844–3852.

[29] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, CoRR abs/1609.02907 (2016). arXiv:1609.02907.

[30] F. Wu, T. Zhang, A. H. S. Jr., C. Fifty, T. Yu, K. Q. Weinberger, Simplifying graph convolutional networks, volume 97 of *Proceedings of Machine Learning Research*, 2019, pp. 6861–6871. arXiv:1902.07153.