Restricted Deep Belief Networks for Multi-view Learning

Yoonseop Kang[1] and Seungjin Choi[1,2]

[1] Department of Computer Science
[2] Division of IT Convergence Engineering
Pohang University of Science and Technology
San 31 Hyoja-dong, Nam-gu, Pohang 790-784, Korea
{e0en,seungjin}@postech.ac.kr

Abstract. Deep belief network (DBN) is a probabilistic generative model with multiple layers of hidden nodes and a layer of visible nodes, where parameterizations between layers obey harmonium or restricted Boltzmann machines (RBMs). In this paper we present *restricted deep belief network* (RDBN) for multi-view learning, where each layer of hidden nodes is composed of *view-specific* and *shared* hidden nodes, in order to learn individual and shared hidden spaces from multiple views of data. View-specific hidden nodes are connected to corresponding view-specific hidden nodes in the lower-layer or visible nodes involving a specific view, whereas shared hidden nodes follow inter-layer connections without restrictions as in standard DBNs. RDBN is trained using layer-wise contrastive divergence learning. Numerical experiments on synthetic and real-world datasets demonstrate the useful behavior of the RDBN, compared to the multi-wing harmonium (MWH) which is a two-layer undirected model.

1 Introduction

Multi-view learning refers to methods for learning from examples that have multiple (independent or dependent) representations, each of which may arise from different views such as modalities or sensors. For instance, in web page classification, one view describes a web page using the text appearing on the document itself, while the other view leads to the anchor text to hyperlinks pointing to this page from other pages [2]. In multimedia mining, images are described by color histograms (one view) and annotated text (the other view), so it is desirable to exploit these two information sources together to boost the performance of image classification [18]. In brain wave classification where EEG data are measured from multiple subjects who undergo the same mental task, each view corresponds to each subject, then learning shared latent spaces provides useful semantic features for EEG classification [11].

Learning latent spaces that capture the relevant information shared by multiple views of data lies at the heart of multi-view learning, especially when views are dependent. One of the oldest but the most popular method is canonical correlation analysis (CCA) [8] which identifies linear relationships between two sets

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 130–145, 2011.

of observations. The shared Gaussian process latent variable model (sGPLVM) [14] is a nonlinear extension of CCA, in which Gaussian process regression is used to learn common hidden structure shared between corresponding sets of heterogenous observations. Manifold integration [4] combines similarity matrices (each of which is determined by a specific view) into a compromise matrix that faithfully reflects multiple sensory information.

Most of these methods assume that views are fully independent conditioned on common latent variables, i.e., only shared latent variables are considered to explain the dependency between views. However, in real-world problems, this assumption is not satisfied. Thus it was suggested that separate latent spaces are learned to model the shared characteristics across views and individual components of observations [11,13]. Group nonnegative matrix factorization (GNMF) learn jointly individual latent spaces and shared latent spaces, decomposing nonnegative data matrix into a product of two nonnegative factor matrices where the basis matrix is composed of common and individual basis vectors [11]. Factorized orthogonal latent space (FOLS) method [13] factorizes the latent space into shared and private latent spaces, enforcing orthogonality between them. These aforementioned methods have limitations since CCA and GNMF are linear techniques and FOLS requires large memory storage and expensive time complexity to handle operations involving a big Gram matrix, which is not scalable as the number of samples increases.

Multi-wing harmonium (MWH) is a two-layer undirected graphical model designed to handle multiple view data. MWH consists of two or more wings where each wing takes the input associated with a single view, extending harmoniums [16] and restricted Boltzmann machines (RBMs) [5]. Harmoniums were also extended to the exponential family [17]. MWH inherits the advantages of exponential family harmoniums (EFHs) such as easy inference and distributed representations over latent variables (see [17] for other advantages). Deep belief network (DBN) is composed of RBMs with multiple layers of hidden nodes, which is trained efficiently in layer-wise manner based on contrastive divergence [5,6]. See [1] for excellent tutorial on DBNs. Multilayer structure of DBNs allow more complex representation of data than RBMs. However, unlike other Boltzmann machine-based models, MWHs cannot be naturally extended to form a deep network.

In this paper we present *restricted deep belief network* (RDBN) for partially correlated multiple view data. We first present a modification of EFHs where view-specific hidden nodes are restricted to have undirected connections to only visible nodes involving corresponding views whereas shared hidden nodes are connected to all the visible nodes. This model inherits advantages of FOLS and MWH simultaneously. The model can efficiently evaluate latent variables as MWH, while still being capable of modelling shared and private information separately.

Then we stack these modified harmoniums to construct RDBN in such a way that view-specific hidden nodes are connected to corresponding view-specific hidden nodes in the lower-layer and shared hidden nodes are connected to all the

hidden nodes in the lower-layer. We train RDBNs using layer-wise contrastive divergence learning, to learn view-specific and shared hidden spaces from multiple views. Numerical experiments on a synthetic dataset, NORB-small dataset, and ESL photo dataset demonstrated the useful behavior of RDBN, compared to MWHs and its direct multilayer version.

2 Related Work

2.1 Exponential Family Harmonium

Most of graphical models including with a layer of hidden nodes and other layer of observed nodes are based on a directed graph. These directed two-layer models allows easy sampling of visible layer, and easy handling of latent variables. However, it is often very difficult and time consuming to calculate posterior distribution of hidden nodes given visible nodes. For the tasks that requires fast evaluation of latent variable given a test sample, directed models would not be very effective.

EFH is a two-layer probabilistic graphical model whose probability distributions are constrained to be exponential family distribution. EFH consists of a set of hidden nodes $\boldsymbol{h}$, and a set of visible nodes $\boldsymbol{v}$ connected by undirected edges. By incorporating undirected connections, the model calculates posterior distribution of hidden nodes $p(\boldsymbol{h}|\boldsymbol{v})$ much faster than directed models, making it more suitable for the tasks including document searching and automatic image annotation.

To define an EFH, we start from choosing marginal distributions for visible nodes $\boldsymbol{v}$ and hidden nodes $\boldsymbol{h}$. As the distributions are constrained to be exponential family distributions, the marginal distributions are defined as below:

$$p(\boldsymbol{v}) = \prod_i \exp\Big\{\sum_a \lambda_{ia} f_{ia}(x_i) - A_i(\{\lambda_{ia}\})\Big\}, \tag{1}$$

$$p(\boldsymbol{h}) = \prod_j \exp\Big\{\sum_c \eta_{jc} g_{jc}(h_j) - B_j(\{\eta_{jc}\})\Big\}, \tag{2}$$

where f_{ia} and g_{jc} are a, and cth sufficient statistics of v_i and h_j. λ^x and η are parameters, and A and B are log partition functions. Note that each v_is and h_j are assumed to be independent to each other.

With this definition, we define joint distribution of $\boldsymbol{v}$ and $\boldsymbol{h}$ by combining their distributions multiplicatively. The joint distribution is defined by introducing quadratic terms to inter-layer relationship:

$$p(\boldsymbol{v}, \boldsymbol{h}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda_{ia} f_{ia}(v_i) + \sum_{j,c} \eta_{jc} g(h_j) + \sum_{i,j,a,c} W_{ijac} f_{ia}(v_i) g_{jb}(h_j)\Big\}. \tag{3}$$

As the model is a bipartite graph, between-layer conditional distributions can be represented as products of distributions of individual nodes. The conditional

distributions are derived as below:

$$p(\boldsymbol{v}|\boldsymbol{h},\theta) = \prod_i \exp\Big\{\sum_a \hat{\lambda}_{ia} f_{ia}(v_i) - A_i(\{\hat{\lambda}_{ia}\})\Big\}, \tag{4}$$

$$p(\boldsymbol{h}|\boldsymbol{v},\theta) = \prod_j \exp\Big\{\sum_c \hat{\eta}_{jc} g_{jc}(h_j) - B_j(\{\hat{\eta}_{jc}\})\Big\}, \tag{5}$$

where shifted parameters are

$$\hat{\lambda}_{ia} = \lambda_{ia} + \sum_{j,c} W_{ijac} g_{jc}(h_j), \tag{6}$$

$$\hat{\eta}_{jc} = \eta_{jc} + \sum_{i,a} W_{ijac} f_{ia}(x_i). \tag{7}$$

As the conditional distribution of nodes are independent to each other, sampling posterior distribution of hidden nodes is done by just simply evaluating conditional distribution $p(\boldsymbol{h}|\boldsymbol{v},\theta)$. An EFH with binary distributions for $\boldsymbol{v}$ and $\boldsymbol{h}$ becomes equivalent to a RBM.

2.2 Multi-Wing Harmonium

Multi-wing harmonium (MWH) [18] models joint distribution of multi-view data using two-layer graphical model. Given two-view data, MWH uses two sets of visible nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ connected to hidden nodes $\boldsymbol{h}$. By choosing different distribution for different type of information, the model achieves better representation of data.

Construction of an MWH is similar to the one of EFH. The only difference is that we split visible nodes $\boldsymbol{v}$ to two sets $\boldsymbol{x}$ and $\boldsymbol{y}$. First we choose marginal distributions for $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{h}$ as below:

$$p(\boldsymbol{x}) = \prod_i \exp\Big\{\sum_a \lambda^x_{ia} f^x_{ia}(x_i) - A^x_i(\{\lambda^x_{ia}\})\Big\}, \tag{8}$$

$$p(\boldsymbol{y}) = \prod_k \exp\Big\{\sum_b \lambda^y_{kb} f^y_{kb}(y_k) - A^y_k(\{\lambda^y_{kb}\})\Big\}, \tag{9}$$

$$p(\boldsymbol{h}) = \prod_j \exp\Big\{\sum_c \eta_{jc} g_{jc}(h_j) - B_j(\{\eta_{jc}\})\Big\}, \tag{10}$$

where f^x_{ia} and f^y_{kb} are a and bth sufficient statistics of x_i and y_k. λ^x, λ^y and η are parameters, and A^x, A^y, and B are log partition functions. Given marginal distribution, joint distribution of nodes is defined straightforwardly:

$$p(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) + \sum_{j,c} \eta_{jc} g(h_j) + \sum_{i,j,a,c} W^x_{ijac} f^x_{ia}(x_i) g_{jb}(h_j) + \sum_{k,j,b,c} W^y_{kjbc} f^y_{kb}(x_y) g_{jc}(h_j)\Big\}. \tag{11}$$

The conditional distributions are derived as below:

$$p(\boldsymbol{x}|\boldsymbol{h},\theta) = \prod_i \exp\Big\{\sum_a \hat{\lambda}^x_{ia} f^x_{ia}(x_i) - A^x_i(\{\hat{\lambda}^x_{ia}\})\Big\}, \tag{12}$$

$$p(\boldsymbol{y}|\boldsymbol{h},\theta) = \prod_k \exp\Big\{\sum_b \hat{\lambda}^y_{kb} f^y_{kb}(y_k) - A^y_k(\{\hat{\lambda}^y_{kb}\})\Big\}, \tag{13}$$

$$p(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{y},\theta) = \prod_j \exp\Big\{\sum_c \hat{\eta}_{jc} g_{jc}(h_j) - B_j(\{\hat{\eta}_{jc}\})\Big\}, \tag{14}$$

with shifted parameters

$$\hat{\lambda}^x_{ia} = \lambda^x_{ia} + \sum_{j,c} W^x_{ijac} g_{jc}(h_j), \tag{15}$$

$$\hat{\lambda}^y_{kb} = \lambda^y_{kb} + \sum_{j,c} W^y_{kjbc} g_{jc}(h_j), \tag{16}$$

$$\hat{\eta}_{jc} = \eta_{jc} + \sum_{i,a} W^x_{ijac} f_{ia}(x_i) + \sum_{k,b} W^y_{kjbc} f_{jb}(y_j). \tag{17}$$

The model can be extended to the case of more than two sets of visible nodes. With a single set of visible nodes, the model shrinks down to a EFH or RBM. To enhance discriminative performance of MWH, labeled version [19] and large-margin approach [3] were also proposed.

3 Restricted DBNs

3.1 Multi-view Harmonium

Many multi-view algorithms are based on a weak assumption that all views are completely correlated. MWH also assumes that views are completely independent when the values of hidden nodes are given. However, the views are often incompletely correlated on real-world datasets. On these datasets, MWH will mix view-specific information with shared information and fail to obtain optimal representation of data. To overcome this problem, we need to model the view-specific information and shared information seperately.

Recent multi-view learning algorithms including work of Salzmann et al.[13] learns from partially independent multi-view data sets by seperating view-specific latent variables from latent variables shared among views.

However, the limitation of existing models is evident. For example, as the FOLS framework requires a Gram matrix, so the model consumes memory storage proportional to N^2, where N is the number of training samples. The evaluation also requires computing time of complexity $O(N)$, as we need to calculate kernel function for every training sample and a test sample. Moreover, existing models including FOLS and group NMF requires additional parameters to control orthogonality between view-specific and shared latent spaces, and these parameters should be selected by going through computationally expensive procedures including cross-validation or by hand.

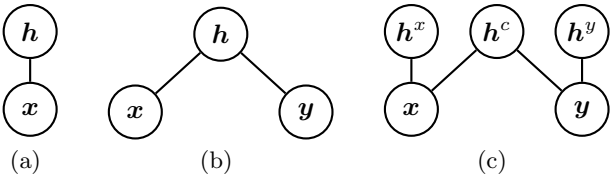

Fig. 1. Graphical models of (a) EFH, (b) MWH, and (c) multi-view harmonium. Repetitions of variables are not denoted for simplicity.

Taking advantage of fast learning and inference of harmonium models and the idea of seperating shared and common latent variables, we extend MWH to devise a new model named *multi-view harmonium*.

This model incorporates view-specific hidden nodes $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ in addition to common hidden nodes $\boldsymbol{h}^c$, to model view-specific, uncorrelated information. $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ are only connected to their corresponding set of visible nodes $\boldsymbol{x}$ and $\boldsymbol{y}$ with connection weights $\boldsymbol{U}^x$ and $\boldsymbol{U}^y$. Graphical representations show the difference between other models and our model (Fig. 1). Joint probability of visible and hidden nodes is as below:

$$
\begin{aligned}
&p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}^x, \boldsymbol{h}^y, \boldsymbol{h}^c | \theta) \propto \\
&\exp\Big\{ \sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) + \sum_{j,c} \eta^c_{jc} g^c(h^c_j) + \sum_{m,d} \eta^x_{md} g^x(h^x_m) \\
&+ \sum_{n,e} \eta^y_{ne} g^y(h^y_n) + \sum_{i,j,a,c} W^x_{ijac} f^x_{ia}(x_i) g_{jb}(h_j) + \sum_{k,j,b,c} W^y_{kjbc} f^y_{kb}(y_k) g_{jc}(h_j) \\
&+ \sum_{i,m,a,d} U^x_{imad} f^x_{ia}(x_i) g^x_{md}(h^x_m) + \sum_{k,n,b,e} U^y_{knbe} f^y_{kb}(y_k) g^y_{ne}(h_n) \Big\}. \qquad (18)
\end{aligned}
$$

Conditional distributions can easily be derived from this joint distribution.

Depending on the type of data, we can choose appropriate distributions from exponential-family. To handle continuous-valued inputs, one can use Gaussian distribution for visible nodes $\boldsymbol{x}$ [7], where the conditional probability is defined as:

$$
p(x_i | \boldsymbol{h}^x, \boldsymbol{h}^c, \theta) = \mathcal{N}(x_i | \boldsymbol{W}^x_i \boldsymbol{h}^c + \boldsymbol{U}^x_i \boldsymbol{h}^x + b^x_i, 1), \qquad (19)
$$

assuming x_i has zero mean and unit variance. Rectified linear units(ReLU) for hidden nodes is also helpful in handling continuous values [12], where value of hidden node h_j given visible node $\boldsymbol{x}$ is defined as:

$$
h_j = \max(0, \sum_i W_{ij} x_i + h_j + \mathcal{N}(0, 1)). \qquad (20)
$$

When modeling term occurence on bag-of-words representation, we can use Poisson distribution:

$$
p(x_i | \boldsymbol{h}^x, \boldsymbol{h}^c, \theta) = \text{Poission}(x_i | \exp(\alpha_i + \boldsymbol{W}^x_i \boldsymbol{h}^x + \boldsymbol{U}^x_i \boldsymbol{h}^c)). \qquad (21)
$$

We train multi-view harmonium by maximizing log-likelihood. Given the joint distribution $p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}^x, \boldsymbol{h}^y, \boldsymbol{h}^c|\theta)$ we can integrate out the hidden units to obtain likelihood function:

$$p(\boldsymbol{x}, \boldsymbol{y}|\theta) \propto \exp\Big\{\sum_{i,a} \lambda^x_{ia} f^x_{ia}(x_i) + \sum_{k,b} \lambda^y_{kb} f^y_{kb}(y_k) - \sum_j B^c(\{\hat{\eta}^c_{jc}\}) - \sum_m B^x(\{\hat{\eta}^x_{md}\}) - \sum_n B^y(\{\hat{\eta}^y_{ne}\})\Big\}, \tag{22}$$

where B^x, B^y and B^c are log-partition functions of marginal distribution of hidden variables $\boldsymbol{h}^c$, $\boldsymbol{h}^x$ and $\boldsymbol{h}^y$ and the shifted parameters are

$$\hat{\eta}^c_{jc} = \eta^c_{jc} + \sum_{i,a} W^x_{ijac} f^x_{ia}(x_i) + \sum_{k,b} W^y_{kjbc} f^y_{kb}(y_j), \tag{23}$$

$$\hat{\eta}^x_{md} = \eta^x_{md} + \sum_{i,a} U^x_{imad} f^x_{ia}(x_i), \tag{24}$$

$$\hat{\eta}^y_{ne} = \eta^y_{ne} + \sum_{k,n,b,e} U^y_{knbe} f^y_{kb}(y_k). \tag{25}$$

To maximize expected log-likelihood over data distribution $\mathcal{L} = \langle \log p(\boldsymbol{x}, \boldsymbol{y}|\theta)\rangle_{\tilde{p}}$, we can use gradient ascent to update parameters θ. The derivatives on parameters related to $\boldsymbol{x}$ are given as follows:

$$\frac{\partial \mathcal{L}}{\partial W^x_{ijac}} = \langle f_{ia}(x_i) B'_{jc}(\hat{\eta}^c_{jc})\rangle_{\tilde{p}} - \langle f_{ia}(x_i) B'_{jc}(\hat{\eta}^c_{jc})\rangle_p, \tag{26}$$

$$\frac{\partial \mathcal{L}}{\partial U^x_{imad}} = \langle f_{ia}(x_i) B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_{\tilde{p}} - \langle f_{ia}(x_i) B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_p, \tag{27}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda^x_{ia}} = \langle f^x_{ia}(x_i)\rangle_{\tilde{p}} - \langle f^x_{ia}(x_i)\rangle_p, \tag{28}$$

$$\frac{\partial \mathcal{L}}{\partial \eta^c_{jc}} = \langle B^{c'}_{jc}(\hat{\eta}^c_{jc})\rangle_{\tilde{p}} - \langle B^{c'}_{jc}(\hat{\eta}^c_{jc})\rangle_p, \tag{29}$$

$$\frac{\partial \mathcal{L}}{\partial \eta^x_{md}} = \langle B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_{\tilde{p}} - \langle B^{x'}_{md}(\hat{\eta}^x_{md})\rangle_p. \tag{30}$$

where $\langle\cdot\rangle_{\tilde{p}}$ and $\langle\cdot\rangle_p$ are expectation over data distribution and model distribution, and $B^{c'}_{jc} = \partial B^c_{jc}/\partial \eta^c_{jc}$ and $B^{x'}_{md} = \partial B^x_{md}/\partial \eta^x_{md}$ are partial derivatives of log-partition functions. Derivatives related to $\boldsymbol{y}$ can be derived similarly. To calculate the expectations from data and model distribution, we need samples from the distributions. However, as we cannot directly sample from the joint distribution of visible nodes and hidden nodes, we use Gibbs sampling and sample from conditional distributions in each Gibbs steps. Values of each node can be sampled from conditional distribution in a single step when other layer's nodes are given.

However, we need infinite steps of Gibbs sampling to calculate exact model distribution. To avoid the problem, we may approximate log-likelihood by using contrastive divergence (CD) learning instead [5]. The key idea of CD learning is

Algorithm 1. Training multi-view harmonium using CD learning with mini-batch size K and M Gibbs steps

Input: $\mathcal{D} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}$
Output: $\theta = \{\boldsymbol{W}^x, \boldsymbol{W}^y, \boldsymbol{U}^x, \boldsymbol{U}^y, \eta^x, \eta^y, \eta^c, \lambda^x, \lambda^y\}$
procedure CDLEARNING($\mathcal{D}$)
    Randomly initialize parameters θ.
    repeat
        Pick K samples as a mini-batch
        repeat
            Sample $\tilde{\boldsymbol{h}}^x \sim p(\boldsymbol{h}^x|\boldsymbol{x}_t, \theta)$, $\tilde{\boldsymbol{h}}^y \sim p(\boldsymbol{h}^y|\boldsymbol{y}_t, \theta)$.
            Sample $\tilde{\boldsymbol{h}}^c \sim p(\boldsymbol{h}^c|\boldsymbol{x}_t, \boldsymbol{y}_t, \theta)$.
            repeat
                Sample $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{h}^x, \boldsymbol{h}^c, \theta)$.
                Sample $\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{h}^y, \boldsymbol{h}^c, \theta)$.
                Sample $\boldsymbol{h}^x \sim p(\boldsymbol{h}^x|\boldsymbol{x}, \theta)$.
                Sample $\boldsymbol{h}^y \sim p(\boldsymbol{h}^y|\boldsymbol{y}, \theta)$.
                Sample $\boldsymbol{h}^c \sim p(\boldsymbol{h}^c|\boldsymbol{x}, \boldsymbol{y}, \theta)$.
            until M steps
        until for all $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ in mini-batch.
        update θ using gradient ascent on $\mathcal{L}$.
    until θ is converged
end procedure

to initialize Gibbs chain to the training data, and to run just a few Gibbs steps instead of infinite steps to approximate model distribution. Even running just a single Gibbs step is known to work well in practice.

Training harmoniums using CD learning requires a gradient calculated over whole training set, and this is often a time consuming task. Instead, we can pick mini-batches from training set, and calculate gradient over mini-batches to save time and space required for training. Summary of procedure for training multi-view harmonium using CD learning is shown in Algorithm 1.

3.2 Restricted DBN

Introducing view-specific hidden nodes brings our model capability of modeling partial correlation, but view-specific hidden nodes also enables us to easily extend our model to form a deep network. Values of view specific hidden layer nodes will be fed as data for upper layer of deep network (Fig. 2). By forming deep network in this manner, the information considered to be "uncorrelated" by limited representation power of current layer will be sent to upper layer and view-to-view correlation will be further analyzed.

Training a deep network using back-propagation is known to be often inefficient. Therefore, training an RDBN is done in greedy, layer-wise manner [6]. We train each layer of multi-view harmoniums using algorithm 1, starting from the bottom layer. After training each layer is finished, a harmonium on the next layer uses samples of view-specific hidden nodes from the trained harmonium

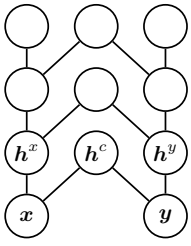

Fig. 2. The graphical model of RDBN. View-specific hidden nodes act as new visible nodes for upper layer of the deep network model.

as its training set. Training of RDBN ends when the procedure reaches the top layer.

3.3 Inferring One View from the Other

Inferring values of unobserved views from other observed views on RDBN is not as easy as inferring the values from two-layer MWH. Instead of using Variational approximation, we choose to perform Gibbs sampling in a systematic manner. The procedure can be divided into three steps, and each step is executed layer by layer (Fig. 3).

1. In the first step, we start from the bottom layer. Given observed values $\boldsymbol{x}$, the hidden nodes of observed view $\boldsymbol{h}^x$ are only nodes that whose probability can be exactly calculated. Therefore we sample the hidden nodes $\boldsymbol{h}^x$ and proceed to the upper layer to repeat this procedure (Fig. 3-b).
2. When the first step reaches the top layer, we run Gibbs sampling for top two layers, while nodes of observed views $\boldsymbol{v}^x$, $\boldsymbol{h}^x$ are fixed to values determined on the first step. In other words, we sample from $p(\boldsymbol{v}^y, \boldsymbol{h}^c, \boldsymbol{h}^y | \boldsymbol{v}^x, \boldsymbol{h}^x, \theta)$. Then we get the values for top two layers of unobserved views $\boldsymbol{h}^c$, $\boldsymbol{v}^y$. We use average of repeatedly sampled values for the unobserved nodes (Fig. 3-c).
3. Finally, we move to lower layers. Given observed nodes $\boldsymbol{x}$ and $\boldsymbol{h}^x$, and hidden units of unobserved views sampled in previous steps $\boldsymbol{h}^y$, we sample from $p(\boldsymbol{y}, \boldsymbol{h}^c | \boldsymbol{x}, \boldsymbol{h}^x, \boldsymbol{h}^y, \theta)$. Repeating the process until we reach bottom layer gives us the values of unobserved views (Fig. 3-d).

When common hidden nodes $\boldsymbol{h}^c$ of bottom layer of deep network are binary logistic nodes, we can skip Gibbs sampling and directly evaluate $p(\boldsymbol{y}|\boldsymbol{x}, \theta)$ up to a normalization constant by rearranging terms:

$$p(\boldsymbol{y}|\boldsymbol{x}, \theta) \propto \prod_j \Big[1 + \exp\Big\{\sum_{i,a}(W^x_{ija} f^x_a(x_i) + \lambda^x_{ia} f^x_a(x_i)) + \sum_{k,b}((W^y_{kjb} + U^y_{knb}) f^y_b(y_k) + \lambda^y_{kb} f^y_b(y_k)) + c^c_j\Big\}\Big], \tag{31}$$

For other layers of RDBN, we use Gibbs sampling instead. Empirically, about 10 times of Gibbs sampling for each layer gave a fair result.

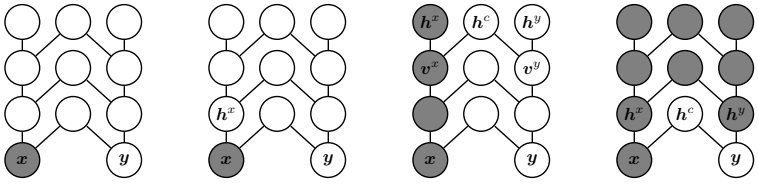

Fig. 3. Inferring procedure of RDBN. Variables with known values at current step is marked as a shaded circle.

4 Numerical Experiments

In this section, we compare our model to MWH. We also compare our method with FOLS-GPLVM [13], and shared GPLVM(SGPLVM) [14], which are directed, non-parametric multi-view latent variable models based on Gaussian process latent variable model [9]. We considered CCA as a baseline too. On a synthetic dataset, we show the effect of modelling view-specific information. Then we run experiments on widely-used NORB-small dataset, and ESL photo dataset in a view-to-view conversion task to numerically compare performance of RDBN and MWH and GPLVM-based models. We used Gibbs sampling for inference on the graphical models.

4.1 Synthetic Example

To show the effectiveness of additional hidden nodes, we performed an experiment taken from recent work of Salzmann et al. [13]. We constructed a synthetic partially correlated multi-view dataset. To illustrate common and view-specific latent variables, we used sinusoidal functions of $\boldsymbol{t}$ with different phases and frequencies:

$$\boldsymbol{x} = \sin(2\pi\boldsymbol{t}), \quad \boldsymbol{z}^1 = \cos(\pi\pi\boldsymbol{t}), \quad \boldsymbol{z}^2 = \sin(\sqrt{5}\pi\boldsymbol{t}),$$
$$\boldsymbol{m}^1 = [\boldsymbol{x}, \boldsymbol{z}^1], \quad \boldsymbol{m}^2 = [\boldsymbol{x}, \boldsymbol{z}^2].$$

We ramdomly projected $\boldsymbol{m}^1$ and $\boldsymbol{m}^2$ to 20 dimensional space and added independent Gaussian noise of variance 0.01 and correlated noise $0.02\sin(3.6\pi\boldsymbol{t})$ to obtain our final multi-view synthetic dataset.

A multi-view harmonium, which is a two-layer RDBN, with 1 common hidden node and 1 view-specific hidden node is used, and we used a MWH and CCA for comparison. 3 hidden nodes for MWH and projection to 3-dimensional space for CCA was chosen for fair comparison. We used Gaussian nodes and ReLU for visible and hidden nodes.

Training was done using 4000 training samples and we calculated hidden node activations from 500 test samples. We checked the correspondence between ground-truth latent variables, and obtained hidden node activations.

MWH, SGPLVM and CCA failed to infer latent variables correctly. No hidden node activations corresponded to latent variables used to generate data. In the

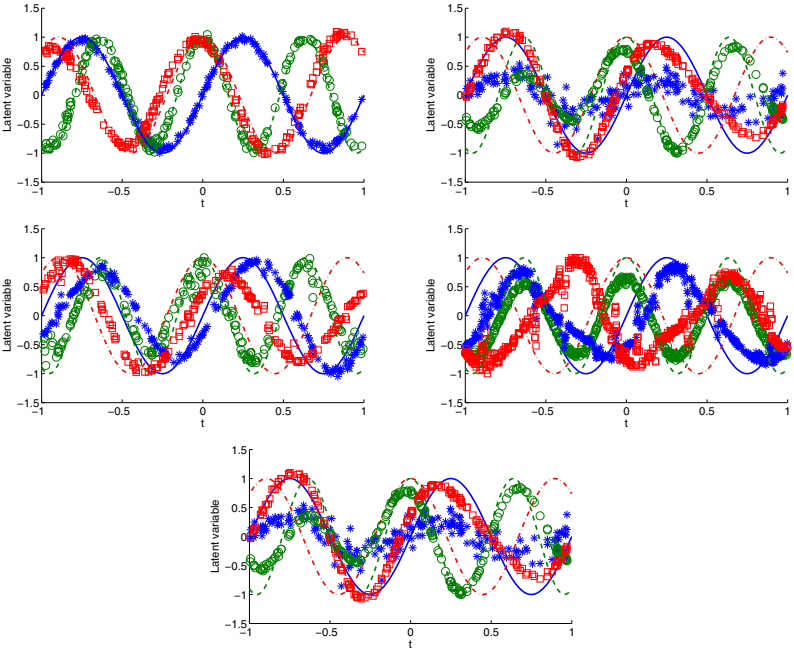

Fig. 4. Latent variables generated by muti-view harmonium , MWH, FOLS, Shared GPLVM and CCA (starting from top-left to bottom-right). Outputs of methods are normalized to have zero mean and maximum absolute value to be 1. Latent variables used to generate data are depicted as lines: Blue solid line: $\boldsymbol{x}$, green dotted line: $\boldsymbol{z}^1$, and red dash-dotted line: $\boldsymbol{z}^2$. Each dimension of outputs are depicted as markers with different colors and shapes.

other hand, our model, multi-view harmonium, found common and view-specific latent variables by its hidden node activations 4. Common hidden nodes and view-specific hidden node activations exactly corresponded to the common and view-specific latent variables. Although FOLS-GPLVM also discovered latent variables correctly, the result was not as accurate as the result of our model. Moreover, our model was able to separate common and view-specific information without any help of additional scheme, while FOLS had to minimize mutual information explicitly.

4.2 Object Conversion on NORB-Small

Training set of NORB-small dataset [10] contains 24,300 stereo images of 5 generic categories of objects on a white background, imaged under various lighting conditions and camera angles. Each category contains 5 instances of objects. For example, the category 'car' contains pictures of 5 different cars.

To evaluate view-to-view conversion performance, we constructed a multi-view dataset by taking pairs of images with same lighting and camera angles, but different category of objects (airplanes - cars), that means each sample in

Fig. 5. Graphical representation of view to view conversion task on NORB-small dataset

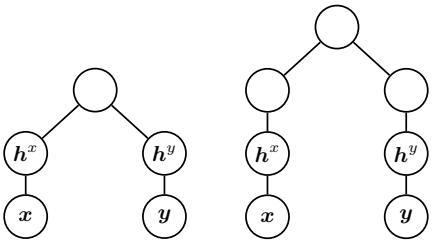

Fig. 6. Graphical model of 3 and 4-layer MWHs used in experiments on NORB-small dataset and ESL dataset

the dataset contains a image of airplane and a car, taken with same condition. 1200 image pairs were used as a training set and 3600 image pairs were used as a test set. After training, model was asked to generate a car image with same imaging condition from a given airplane image (Fig. 5). Images were resized to the size 32×32, and pixels on each position of an image were normalized to have zero mean and variance 1. The variance of pixel values was calculated across whole training set on the same position on images. The root mean squared error (RMSE) was computed:

$$\text{RMSE} = \sqrt{\frac{1}{N_{samples}} \sum_t \frac{1}{N_{pixels}} ||\boldsymbol{x}_t^{truth} - \boldsymbol{x}_t^{rec}||^2}, \tag{32}$$

where $N_{samples}$ and N_{pixels} are the number of samples and of pixels, respectively, and $\boldsymbol{x}_t^{truth}$ and $\boldsymbol{x}_t^{rec}$ are ground-truth and reconstructed images, respectively.

For our experiment, we constructed 2, 3, and 4-layer RDBN. For two layers from the bottom, we used Gaussian and ReLU, to handle continuous valued data. Logistic binary nodes were used for remaining two layers.

We also constructed a deep network model for MWH by attaching harmoniums under each views of MWH for a fair comparison with RDBN. For example, 4-layer MWH was constructed by attaching additional two layers of Harmoniums under each view of a MWH (Fig. 6). Inferring an unobserved view was done by the procedure below.

1. Sample nodes of observed view layer by layer from the bottom.
2. Infer the unobserved view of MWH on the top layer.
3. Sample unobserved nodes, from top to bottom layer.

Table 1. Averaged RMS reconstruction errors of RDBN, MWH models and random reconstruction in the view-to-view conversion task on NORB-small dataset

Method	RMSE	Method	RMSE
4-layer RDBN	**0.0923**	4-layer MWH	0.1379
3-layer RDBN	0.0936	3-layer MWH	0.1377
2-layer RDBN	0.0958	2-layer MWH	0.0973
FOLS-GPLVM	0.2489	SGPLVM	0.2089
Random	0.5205		

We also added additional hidden nodes to MWH to ensure same number of parameters are used. For example, if a RDBN had 200 view-specific hidden nodes and 800 common hidden nodes, each view of corresponding layer of a MWH had $200 + 800 = 1000$ hidden nodes. We also tried 2-layer and 3-layer RDBN to see the effect of number of layers. Each network was trained 200 iterations with learning rate of 0.001. Instead of full batch learning, we used mini-batches with 100 samples. SGPLVM and FOLS-GPLVM were trained with 200 iterations, and their latent dimension was automatically decided by a heuristic method of Neil Lawrence's software. As a baseline, we used images with random pixel values as reconstruction images and compared the reconstruction errors with the results of RDBNs and MWH models.

The experimental result showed remarkable difference between RDBN and MWH algorithm. 4-layer RDBN showed the smallest error than any configuration of MWH, FOLS-GPLVM and SGPLVM. Moreover, adding each additional layer to RDBN reduced reconstruction error, while adding layer to MWH damaged the performance of the model (Table 1).

4.3 Image Annotation on ESL Photo Dataset

To examine the capability of our deep model to find nontrivial relation between views, we applied our model to the task of image annotation. The experiment was done using ESL photo dataset [15], which contains 3464 photos collected from Flickr, annotated with 59 different tags. We calculated autocorrelogram of images with the setting of radius 6 and 32 colors on HSV color space. By this process we obtained a 192-dimensional feature vector for each image. We also calculated TF-IDF from our tag data. We used 2000 images for training, and 1044 images testing.

We compared 2, 3, 4-layer RDBNs, 2-layer MWH, sGPLVM, and FOLS-GPLVM. 30, 40, 60 hidden units were used for each hidden layer of RDBNs, and GPLVM models were trained with 200 iterations. Image autocorrelogram and TF-IDF of tags were assigned to each view of the models. On visible layer of the models, we used Gaussian nodes for images and Poisson nodes for tags. Each model was trained 500 iterations with learning rate 0.01. Again, we used mini-batches with 100 samples in training. We also tried annotating images with random tags as a baseline.

Precision-recall curve and mean average precision (MAP) were used as evaluation measure. MAP is defined as the mean of average precision (AP) over test

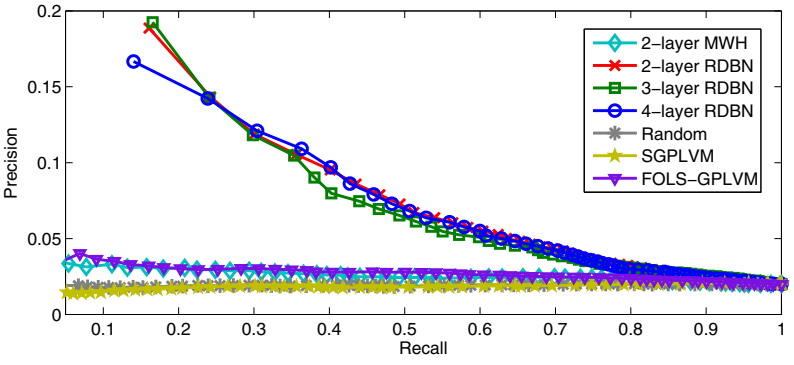

Fig. 7. Precision-recall curves for RDBNs, 2-layer MWH, SGPLVM, FOLS-GPLVM and random annotation. Vertical axis of the plot is trimmed for better comparison between results of RDBNs with different number of layers.

Table 2. The mean average precision of RDBNs, a MWH, SGPLVM, FOLS-GPLVM and random annotation in image annotation task on ESL dataset.

Method	MAP	Method	MAP
4-layer RDBN	**0.0620**	2-layer MWH	0.0251
3-layer RDBN	0.0598	SGPLVM	0.0180
2-layer RDBN	0.0565	FOLS-GPLVM	0.0267
Random	0.0190		

samples:

$$\mathrm{MAP} = \frac{\sum_t \mathrm{AP}(\boldsymbol{x}_t)}{N_{samples}} = \frac{1}{N_{samples}} \sum_t \sum_{j=1}^{N_{tags}} \mathrm{Precision}(\boldsymbol{x}_t, j), \tag{33}$$

where $N_{samples}$ and N_{tags} are the number of samples and of tags, respectively, and $\mathrm{Precision}(\boldsymbol{x}_t, j)$ denotes the precision when top j tags are chosen, given the query image $\boldsymbol{x}_t$. MAP is also calculated by area under precision-recall curve. We used this alternative definition of MAP to calculate the results reported here.

Precision-recall curve shows that RDBN algorithms clearly outperformed other methods, showing the advantage of modelling view-specific information seperately. Although the difference was not very significant, deep models showed higher average precision than 2-layer RDBN, showing the benefit of using deep models. In contrast, average precision of 2-layer MWH and SGPLVM were not very distinguishable from the result of random annotation (Table 2). Precision-recall curve also shows the difference between RDBNs and other models (Fig. 7).

5 Conclusions

In this paper, we have proposed the harmonium-based model that uses view-specific hidden nodes for multi-view data, to capture the view-specific

information of the data separated from shared information among views. Moreover, our model can be naturally extended to deep network to model more complex relationship between views by using view-specific hidden nodes as inputs to next layer. We have demonstrated the effectiveness of our approach by comparing our model to MWH, and their directed, non-parametric counterparts including SGPLVM and FOLS-GPLVM. the existing harmonium-based multi-view model, on various experiments on synthetic and real-world examples. And we showed significant improvement over other methods in the tested examples. In the future, we plan to investigate the modifications of RDBN model such as conditional RDBN to directly model conditional distribution between views. We also plan to extend our model to use label information for supervised and semi-supervised learning. Another future topic is extension of our model to time-series data or other structured data.

Acknowledgments. This work was supported by the Converging Research Center Program funded by the Ministry of Education, Science, and Technology (No. 2010K001171), NIPA ITRC Support Program (NIPA-2011-C1090-1131-0009), NIPA-MSRA Creative IT/SW Research Project, and NRF WCU Program (R31-10100).

References

1. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends® in Machine Learning 2(1), 1–127 (2009)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Annual Conference on Learning Theory (COLT), Madison, Wisconsin (1998)
3. Chen, N., Zhu, J., Xing, E.P.: Predictive subspace learning for multi-view data: A large margin approach. In: Advances in Neural Information Processing Systems (NIPS), vol. 23. MIT Press, Cambridge (2010)
4. Choi, H., Choi, S., Choe, Y.: Manifold integration with Markov random walk. In: Proceedings of the AAAI National Conference on Artificial Intelligence (AAAI), Chicago, IL (2008)
5. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
6. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
7. Hinton, G.E., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
8. Hotelling, H.: Relations between two sets of variates. Biometrika 28, 312–377 (1936)
9. Lawrence, N.: Gaussian process latent variable models for visualisation of high dimensional data. In: Advances in Neural Information Processing Systems (NIPS), vol. 16. MIT Press, Cambridge (2004)
10. LeCun, Y., Huang, F.J., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC (2004)

11. Lee, H., Choi, S.: Group nonnegative matrix factorization for EEG classification. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Clearwater Beach, Florida (2009)
12. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the International Conference on Machine Learning (ICML), Haifa, Israel (2010)
13. Salzmann, M., Ek, C.H., Urtasun, R., Darrell, T.: Factorized orthogonal latent spaces. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Sardinia, Italy (2010)
14. Shon, A.P., Grochow, K., Hertzmann, A., Rao, R.P.N.: Learning shared latent structure for image synthesis and robotic imitation. In: Advances in Neural Information Processing Systems (NIPS), vol. 17. MIT Press, Cambridge (2006)
15. Sinha, P., Jaini, R.: Classification and annotation of digital photos using optical context data. In: Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR), Niagara Falls, Canada (2008)
16. Smolensky, P.: Information processing in dynamical systems: Foundations of harmony theory. In: Rumelhart, D.E., McClelland, J.L. (eds.) Parallel Distributed Processing: Explorations in the Microstructures of Cognition: Foundations, vol. 1, pp. 194–281. MIT Press, Cambridge (1986)
17. Welling, M., Rosen-Zvi, M., Hinton, G.: Exponential family harmoniums with an application to information retrieval. In: Advances in Neural Information Processing Systems (NIPS), vol. 17. MIT Press, Cambridge (2005)
18. Xing, E.P., Yan, R., Hauptmann, A.G.: Mining associated text and images with dual-wing harmonium. In: Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI), Edinburgh, UK (2005)
19. Yang, J., Liu, Y., Xing, E.P., Hauptmann, A.G.: Harmonium models for semantic video representation and classification. In: Proceedings of the SIAM International Conference on Data Mining (SDM), Minineapolis, MN (2007)