Deep Models for Asynchronous Multiview Sequential Learning

Tung DOAN[†] and Atsuhiro TAKASU[†,††]

† School of Multidisciplinary Sciences, The Graduate University for Advanced Studies, SOKENDAI
Shonan Village, Hayama, Kanagawa 240–0193, Japan
†† National Institute of Informatics,
2–1–2 Hitotsubashi, Chiyoda-ku, Tokyo 101–8430, Japan
E-mail: †{tungdp,takasu}@nii.ac.jp

Abstract Multiview representation learning has become an active research topic in machine learning and data mining. One underlying assumption of the conventional methods is that training data of the views must be equal in size and sample–wise matching. However, in many real–world applications, such as video analysis, text streaming, and signal processing, data for the views often come in the form of asynchronous sequences, that are different in length and misaligned. This results in the failure of directly applying existing methods to handle multiview sequential data. In this paper, we first introduce a novel deep multi-view model that can implicitly discover sample correspondence while learning the representation. It can be shown that our method generalizes deep canonical correlation analysis – a popular multiview learning method. We then extend our model by integrating the objective function with the reconstruction losses of autoencoders, forming a new variant of the proposed model. Extensively experimental results demonstrate the superior performances of our models over competing methods.

Key words Multiview learning, dynamic time warping, smooth approximation, deep learning, sequential data

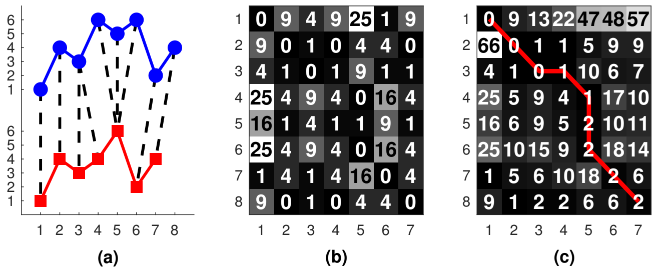
1 Introduction

Multi-view learning methods aim at exploiting complementary information between the views to learn new representations for the data that are more beneficial for tasks such as clustering and classification.

Canonical correlation analysis (CCA) [1] is representative of unsupervised multi-view learning methods. It tries to find a latent subspace in which projections of the views are maximally correlated. Recently, [2], [3] introduced deep CCA (DCCA), which is an extension of CCA based on deep learning. DCCA learns nonlinear mapping functions for the views using deep neural networks (DNNs), whose weights are optimized to maximize the correlation between the outputs. It was shown that CCA problem is equivalent to minimizing the squared difference between the projections of the views, subject to the whitening constraints.

Although CCA and its variants have shown promising performances in different applications, they share a serious limitation in that they are designed under the assumption that training data of the views are equal in size and sample-wise matching. More specifically, CCA-based methods require two-view training data: $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d_x \times n}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, ..., \boldsymbol{y}_n] \in \mathbb{R}^{d_y \times n}$, where $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is a matching pair $(1 \leqq i \leqq n)$. However, these requirements are likely to be violated in practice when training data are often asynchronous sequences. Taking a camera network as an example, multiple cameras capture the same scene from different angles. Because of temporal failure of some cameras and man-made reasons, frame deletion and/or insertion often occur. As a result, the sizes of the obtained video sequences are unequal and the frame-wise correspondence is missing. The misalignment is also likely to happen in text streaming, where documents are possibly collected from different sources, and multi-channel signal processing because the sensors might have dissimilar sampling frequencies. A widely used alignment algorithm, dynamic time warping (DTW) [4], can be used to match samples in correspondence as a preprocessing step before performing conventional multi-view learning methods. Unfortunately, DTW fails when the two-view data are dimensionally different $(d_x \neq d_y)$.

In this paper, we introduce a novel DNN-based method to handle the aforementioned problem. Our method passes data sequences through DNNs to map them into the same subspace. By minimizing the generalized smooth DTW distance – a differentiable approximation of original DTW cost function – between the projections of the two views, our method can implicitly discover the sample correspondence while learning representations. In addition, our method uses soft regularizations to approximate hard whitening constraints in CCA-based methods that force the covariance matrices of the outputs over the training set to be iden-



	1	2	3	4	5	6	7
1	0	9	4	9	25	1	9
2	9	0	1	0	4	4	0
3	4	1	0	1	9	1	1
4	25	4	9	4	0	16	4
5	16	1	4	1	1	9	1
6	25	4	9	4	0	16	4
7	1	4	1	4	16	0	4
8	9	0	1	0	4	4	0

	1	2	3	4	5	6	7
1	0	9	13	22	47	48	57
2	66	0	1	1	5	9	9
3	4	1	0	1	10	6	7
4	25	5	9	4	1	17	10
5	16	6	9	5	2	10	11
6	25	10	15	9	2	18	14
7	1	5	6	10	18	2	6
8	9	1	2	2	6	6	2

Figure 1 An example of DTW: (a) the two sequences, (b) the distance matrix, and (c) the optimal alignment matrix.

tity matrices. This makes our model more suitable to be trained by stochastic gradient descent (SGD) because the objective function is unconstrained. Furthermore, coupling soft whitening constraints with an objective function may allows more globally optimal solutions to be obtained. Our approach can be easily applied to other deep models to improve the performances on multi-view representation learning from sequential data. In this work, we combine our objective function with reconstruction losses of autoencoders [5] to form a new variant of the proposed method.

2 Background

2 1 Dynamic Time Warping

The DTW [4] algorithm measures the similarity between two sequences whose lengths are possibly different and whose sample correspondences are probably unknown. Given two sequences $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, ..., \boldsymbol{y}_m] \in \mathbb{R}^{d \times m}$, a distance matrix $\boldsymbol{D}(\boldsymbol{X}, \boldsymbol{Y}) \in \mathbb{R}^{n \times m}$ is defined such that the element at position (i, j), denoted by $d_{i,j}$, is the squared distance, i.e., $d_{i,j} = ||\boldsymbol{x}_i - \boldsymbol{y}_j||_2^2$. The DTW algorithm constructs a cumulative sum matrix $\boldsymbol{S}(\boldsymbol{X}, \boldsymbol{Y})$ using the following recursive formulas:

$$s_{1,1} = d_{1,1} \tag{1}$$

$$s_{i,j} = d_{i,j} + \min(s_{i-1,j}, s_{i,j-1}, s_{i-1,j-1}), \tag{2}$$

The DTW distance between the two sequences is then defined as $\mathrm{DTW}(\boldsymbol{X}, \boldsymbol{Y}) := s_{n,m}$. By backtracking from the last element $s_{n,m}$ to the start element $s_{1,1}$, an optimal warping path

$$\boldsymbol{\pi}^* = \langle (i_1^*, j_1^*), \ldots, (i_p^*, j_p^*) \rangle, \tag{3}$$

that satisfies: i) *boundary condition* : $(i_1, j_1) = (1, 1)$ and $(i_p, j_p) = (n, m)$; ii) *continuous condition* : $(i_{r+1} - i_r, j_{r+1} - j_r) \in \{(0, 1), (1, 0), (1, 1)\}$, where $1 \leqq r \leqq p - 1$; and iii) *monotonic condition*: if $1 \leqq r \leqq t \leqq p$, then $i_r \leqq i_t$ and $j_r \leqq j_t$ is formed. This path has the smallest cumulative sum $s_{n,m} = d_{i_1^*, j_1^*} + \cdots + d_{i_p^*, j_p^*}$ and encodes the sample correspondences between the two sequences. A toy example of DTW is shown in Figure 1.

2 2 Generalized Smooth DTW

The optimal warping path can be discovered by minimizing DTW; however, original DTW is not differentiable because of the nonsmoothness of min operator in equation (2), which makes it difficult to minimize using gradient-based methods. To alleviate this issue, [6], [7] studied a smooth min operator that serves as an essential basis to develop the differentiable approximations of DTW.

Let $\boldsymbol{\eta} = [\eta_1, ..., \eta_k]^\top \in \mathbb{R}^k$, the smooth min operator is defined as follows:

$$\min\nolimits_\Omega(\boldsymbol{\eta}) := \min_{\boldsymbol{\gamma} \in \Delta^k} \langle \boldsymbol{\gamma}, \boldsymbol{\eta} \rangle + \frac{1}{\beta} \Omega(\boldsymbol{\gamma}), \tag{4}$$

where $\Delta^k := \{\boldsymbol{\gamma} \in \mathbb{R}_+^k : ||\boldsymbol{\gamma}||_1 = 1\}$ is a $(k-1)$ unit simplex, $\langle .,. \rangle$ denotes an inner product, Ω is a strictly convex function on Δ^k, and β is a nonnegative regularization parameter. Because (4) is strictly convex, its minimum is unique and equal to the gradient (based on Danskin's theorem [8]):

$$\nabla \min\nolimits_\Omega(\boldsymbol{\eta}) = \operatorname*{argmin}_{\boldsymbol{\gamma} \in \Delta^k} \langle \boldsymbol{\gamma}, \boldsymbol{\eta} \rangle + \frac{1}{\beta} \Omega(\boldsymbol{\gamma}). \tag{5}$$

The equation shows that the smooth min operator also depends on the selection of the regularization function $\Omega(\boldsymbol{\gamma})$. Shannon entropy $(\sum_{i=1}^k \gamma_i \ln \gamma_i)$ or squared ℓ_2 norm $(\frac{1}{2} \sum_{i=1}^k \gamma_i^2)$ are often chosen. While the former induces closed-form solutions for both smooth min and its gradient, the latter forces the gradient to be sparse.

As the definition of the smooth min operator is already given, we can arrive at the following recursive formulation:

$$s'_{1,1} = d_{1,1}$$

$$s'_{i,j} = d_{i,j} + \min\nolimits_\Omega(s'_{i-1,j}, s'_{i,j-1}, s'_{i-1,j-1}), \tag{6}$$

where the generalized smooth approximation of DTW is defined by $\mathrm{DTW}_\Omega(\boldsymbol{X}, \boldsymbol{Y}) := s'_{n,m}$. Note that we can have different versions of DTW_Ω, e.g., $\mathrm{DTW}_{\Omega=\text{entropy}}$ or $\mathrm{DTW}_{\Omega=\text{squared } \ell_2}$, depending on selection of the regularization $\Omega(\boldsymbol{\gamma})$. The generalized smooth DTW distance is different from the original DTW because it is differentiable. Furthermore, by minimizing DTW_Ω, the optimal warping path is discovered implicitly instead of specified directly, as in the original DTW.

3 The Proposed Method

In this section, we propose a method, namely *deep sequential correlation analysis* (DSCA), for multi-view representation learning from sequential data. We first present the model and its objective function. We then describe the optimization methods and the relationship between the proposed method and DCCA [2].

Given two data sequences $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d_x \times n}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, ..., \boldsymbol{y}_m] \in \mathbb{R}^{d_y \times m}$, our method passes each of them

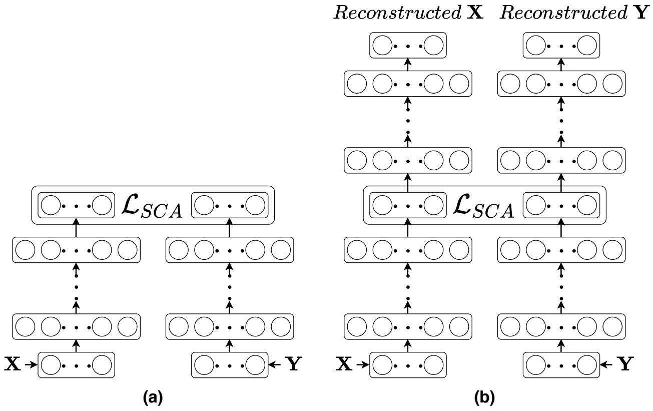


Figure 2 Diagrams of (a) the proposed method and (b) its autoencoder-based variant.

through a DNN to compute their representations. For each DNN, we also add a batch normalization (BN) [9] layer on top so each feature of the representations over the training dataset has zero mean and unit variance. We denote $\boldsymbol{\theta}_x = \{\boldsymbol{W}_x^{(l)}, \boldsymbol{b}_x^{(l)} | 1 \leqq l \leqq l_x\}$ as a collection of all weight matrices and biases of l_x layers in the first DNN. Passing an instance $\boldsymbol{x}_i \in \boldsymbol{X}$ through the first DNN, its output at the top layer is $\boldsymbol{z}_i^x = f_x(\boldsymbol{x}_i, \boldsymbol{\theta}_x) \in \mathbb{R}^d$, where $f_x(\cdot)$ denotes the nonlinear mapping performed by the first DNN and d is the dimension of the representation. By assembling all output vectors $\boldsymbol{z}_i^x$ into the columns of a matrix following the increasing order of the index i, we have a representation sequence of the first view: $\boldsymbol{Z}^x = f_x(\boldsymbol{X}, \boldsymbol{\theta}_x) \in \mathbb{R}^{d \times n}$ (Similarly, $\boldsymbol{Z}^y = f_y(\boldsymbol{Y}, \boldsymbol{\theta}_y) \in \mathbb{R}^{d \times m}$ for the second view). Figure 2 (a) shows the diagram of the proposed method.

3 1 Objective

Our method minimizes the following objective:

$$\mathcal{L}_{SCA} = \mathrm{DTW}_\Omega(\boldsymbol{Z}^x, \boldsymbol{Z}^y) + \lambda_1 \mathcal{L}_x(\boldsymbol{Z}^x) + \lambda_2 \mathcal{L}_y(\boldsymbol{Z}^y), \quad (7)$$

where the two regularization terms are of the following form:

$$\mathcal{L}_v(\boldsymbol{Z}^v) = \sum_{i=1}^{d} \sum_{j \neq i}^{d} |c_{i,j}^v|, \quad (8)$$

where $v \in \{x, y\}$ and $c_{i,j}^v$ is the element at the (i, j) position of the matrix $\boldsymbol{C}^v = \boldsymbol{Z}^v \boldsymbol{Z}^{v\top}$. These regularization functions are soft approximations of the whitening constraints in CCA-based methods. More specifically, the whitening constraints enforce the features of the representations to be pairwise uncorrelated ($\boldsymbol{C}^v = \boldsymbol{I}$). They are used to prevent trivial solutions, e.g., all the data samples are mapped into a single point in the shared subspace. In our method, because the representation sequences are normalized by BN layers, we further use the l_1-norm to encourage sparsity in the off-diagonal elements of $\boldsymbol{C}^v$. $\lambda_1 > 0$ and $\lambda_2 > 0$ are regularization parameters that control the trade-off between whitening and warping the two representation sequences.

Algorithm 1 : Stochastic algorithm for DSCA

```
Require: Batch size ratio $\alpha \in [0,1]$, time constant $\rho \in [0,1]$, momentum $\mu \in [0,1)$, and learning rate $\epsilon$.
Ensure: Optimal DNNs parameters $\boldsymbol{\theta}^* = [\boldsymbol{\theta}_x^*, \boldsymbol{\theta}_y^*]$.
for $t = 1, \dots, \mathrm{T}$ do
    random sample subsequence $\boldsymbol{Z}_{(t)}^x$ of length $n\alpha$;
    random sample subsequence $\boldsymbol{Z}_{(t)}^y$ of length $m\alpha$;
    $\boldsymbol{C}_{(t)}^x = \rho \boldsymbol{C}_{(t-1)}^x + (1-\rho)\frac{1}{\alpha}\boldsymbol{Z}_{(t)}^x \boldsymbol{Z}_{(t)}^{x\ \top}$;
    $\boldsymbol{C}_{(t)}^y = \rho \boldsymbol{C}_{(t-1)}^y + (1-\rho)\frac{1}{\alpha}\boldsymbol{Z}_{(t)}^y \boldsymbol{Z}_{(t)}^{y\ \top}$;
    compute $\frac{\partial \mathcal{L}_{SCA}}{\partial \boldsymbol{Z}_{(t)}^x}$ and $\frac{\partial \mathcal{L}_{SCA}}{\partial \boldsymbol{Z}_{(t)}^y}$;
    compute gradient $\nabla_{\boldsymbol{\theta}}$ using backpropagation;
    $\Delta\boldsymbol{\theta}_{(t)} = \mu \Delta\boldsymbol{\theta}_{(t-1)} - \epsilon \nabla_{\boldsymbol{\theta}}$;
    $\boldsymbol{\theta}_{(t)} = \boldsymbol{\theta}_{(t-1)} + \Delta\boldsymbol{\theta}_{(t)}$;
end for
```

3 2 Optimization

The parameters $\boldsymbol{\theta}_x$ and $\boldsymbol{\theta}_y$ can be trained using the gradient-based method. To compute the gradient of $\mathcal{L}_{SCA}$ w.r.t. all the parameters $\boldsymbol{\theta}_x$ and $\boldsymbol{\theta}_y$, we compute its gradients w.r.t. the outputs $\boldsymbol{Z}^x$ and $\boldsymbol{Z}^y$ and then use backpropagation [10]. We have

$$\frac{\partial \mathcal{L}_{SCA}}{\partial \boldsymbol{Z}^x} = \frac{\partial \mathrm{DTW}_\Omega(\boldsymbol{Z}^x, \boldsymbol{Z}^y)}{\partial \boldsymbol{Z}^x} + \lambda_1 \frac{\partial \mathcal{L}_x(\boldsymbol{Z}^x)}{\partial \boldsymbol{Z}^x}. \quad (9)$$

The gradient of the generalized smooth DTW w.r.t. $\boldsymbol{Z}^x$ can be computed as

$$\frac{\partial \mathrm{DTW}_\Omega(\boldsymbol{Z}^x, \boldsymbol{Z}^y)}{\partial \boldsymbol{Z}^x} = \left[\frac{\partial s'_{n,m}}{\partial \boldsymbol{z}_1^x}, \dots, \frac{\partial s'_{n,m}}{\partial \boldsymbol{z}_n^x}\right], \quad (10)$$

where

$$\frac{\partial s'_{n,m}}{\partial \boldsymbol{z}_i^x} = \sum_{j=1}^{m} \frac{\partial s'_{n,m}}{\partial d_{i,j}} \frac{\partial d_{i,j}}{\partial \boldsymbol{z}_i^x} \quad (11)$$

$$= 2 \sum_{j=1}^{m} e_{i,j} \left(\boldsymbol{z}_i^x - \boldsymbol{z}_j^y\right) \qquad \text{for } i = 1, ..., n. \quad (12)$$

In equation (12), we abused the notations defined in Section 2, where $s'_{n,m} \coloneqq \mathrm{DTW}_\Omega(\boldsymbol{Z}^x, \boldsymbol{Z}^y)$ and $d_{i,j} \coloneqq ||\boldsymbol{z}_i^x - \boldsymbol{z}_j^y||_2^2$. The derivative $e_{i,j} = \frac{\partial s'_{n,m}}{\partial d_{i,j}}$ can be computed efficiently using a forward-backward algorithm [7].

The gradient of $\mathcal{L}_x$ w.r.t. $\boldsymbol{Z}^x$ can be computed as

$$\frac{\partial \mathcal{L}_x(\boldsymbol{Z}^x)}{\partial \boldsymbol{Z}^x} = \boldsymbol{H}^x \boldsymbol{Z}^x, \quad (13)$$

where $\boldsymbol{H}^x \in \mathbb{R}^{d \times d}$, whose elements are defined as

$$h_{i,j}^x = \begin{cases} 1 & \text{if } c_{i,j}^x > 0 \\ 0 & \text{if } i = j \text{ or } c_{i,j}^x = 0 \\ -1 & \text{if } c_{i,j}^x < 0. \end{cases} \quad (14)$$

The gradient $\frac{\partial \mathcal{L}_{SCA}}{\partial \boldsymbol{Z}^y}$ can be computed in a similar manner. Our model can be trained using a full-batch algorithm (L-BFGS) [11], as in [2]. For large datasets, however, this algorithm is both time and memory inefficient. An alternative is based on stochastic gradient descent (SGD) [12], [13] where the gradient is estimated based on a much smaller number of training samples (a minibatch). The details are shown in Algorithm 1. Note that we use a stochastic estimate of the covariance matrix for each view because at each iteration t, the algorithm can access only a small number of samples instead of the whole training set.

3 3 Relation to Deep CCA

Let us consider the case where the data sequences X and Y are equal in size ($m = n$), then the DTW distance between $\boldsymbol{Z}^x$ and $\boldsymbol{Z}^y$ is equivalent to their squared difference. By replacing the two regulation terms with their associated hard whitening constraints, the optimization problem of DSCA turns into

$$\begin{aligned} \min_{\boldsymbol{\theta}_x, \boldsymbol{\theta}_y} \quad & ||\boldsymbol{Z}^x - \boldsymbol{Z}^y||_F^2 \\ \text{s.t.} \quad & \boldsymbol{Z}^x \boldsymbol{Z}^{x\top} = \boldsymbol{Z}^y \boldsymbol{Z}^{y\top} = \boldsymbol{I}, \end{aligned} \tag{15}$$

which is exactly the optimization problem of DCCA. This indicates that DSCA also maximizes the correlation between the projections of the views. However, our method is more generalized than DCCA, since it can handle multi-view sequential data, which are possibly unequal in size and misaligned.

4 Autoencoder Variant

In this section, we extend the proposed model by adding a deep reconstruction branch for each view, forming a model with two autoencoders, which we call *deep sequentially correlated autoencoders* (DSCAE). Diagram of DSCAE is illustrated in Figure 2 (b). Let $\mathrm{g}_x(\cdot)$ and $\mathrm{g}_y(\cdot)$ denotes the transformations performed by the two additional branches and $\boldsymbol{\gamma}_x$ and $\boldsymbol{\gamma}_y$ are their corresponding parameters. The optimization problem of DSCAE is

$$\begin{aligned} \underset{(\boldsymbol{\theta}_x, \boldsymbol{\theta}_y, \boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y)}{min} \mathcal{L}_{SCA} + \lambda \Bigg(& \frac{1}{n} \sum_{i=1}^{n} ||\boldsymbol{x}_i - \mathrm{g}_x(\boldsymbol{z}_i^x, \boldsymbol{\gamma}_x)||^2 \\ & + \frac{1}{m} \sum_{j=1}^{m} ||\boldsymbol{y}_j - \mathrm{g}_y(\boldsymbol{z}_j^y, \boldsymbol{\gamma}_y)||^2 \Bigg), \end{aligned} \tag{16}$$

where $\lambda > 0$ is a trade-off parameter. Since solving for $\mathcal{L}_{SCA}$ allows us to implicitly discover sample correspondence between the views and minimize the squared difference between their projections, it amounts to performing CCA objective, which maximizes the mutual information between the projected views, on sequential data. On the other hand, minimizing the reconstruction errors is equivalent to maximizing a bound on the mutual information between the input and output of each view. Therefore, DSCAE allows a trade-off between information within each view mapping and information in the correlation across the views.

We also use a stochastic-based algorithm to train DSCAE. Note that the gradients for $\boldsymbol{\theta}_x$ and $\boldsymbol{\theta}_y$ are computed as in DSCA and added by the terms associated with the autoencoder parts, while the gradient for $\boldsymbol{\gamma}_x$ and $\boldsymbol{\gamma}_y$ are only dependent on the reconstruction losses.

5 Related Work

As noted, conventional methods such as CCA and its variants can not handle the case where data sequences are unequal in size and misaligned. One approach is to combine them directly with DTW [14], [15], [16] to find a subspace, where projections of the two sequences are aligned and the learned representations of the two views are maximally correlated. However, this direct combination has some serious drawbacks. Since DTW problem is discrete and its objective is not differentiable, the alignment and representations are not optimized in a unified manner. Specifically, they need to be optimized alternately (one is fixed while optimizing the other), resulting in suboptimal solutions. In addition, this approach is much more inefficient when DNNs are used to map the two views into the new subspace as their expensive training procedures need to be performed multiple times. Therefore, this approach is unsuitable for applying to other deep models.

Another approach to multi-view learning without the requirement of sample-wise correspondence is based on unsupervised manifold alignment [17], [18], [19]. This approach maps data samples from two different spaces into a subspace simultaneously matching the samples in correspondence and preserving local geometry of the sets. Nevertheless, these methods are limited to shallow models and sensitive to noise, which corrupts the adjacency and geometric information of the data.

[20], [21] proposed hybrid methods that combine unsupervised manifold alignment with DTW. Thus, they also inherit drawbacks from the two presented approaches. Our methods (DSCA and DSCAE) differ from the first approach because they discover the sample correspondence implicitly while learning the representations instead of alternately updating the projections of the views and aligning them. Thus, more global solutions are expected to be achieved. Futhermore, the proposed objective functions allow us to design an efficient stochastic algorithm for training the models, making them applicable to other deep models. Our method also differs from the second approach because the DNNs are used to learn more robust and richer nonlinear embeddings.

6 Experiments

6 1 Datasets

We first briefly describe the preprocessing steps on three datasets used in the experiments. We then show how to generate misaligned data sequences from these datasets.

Noisy MNIST digits. We generate two-view data from MNIST dataset [13] that consists of 28 × 28 grayscale digit [0—9] images divided into $60K/10K$ for training/testing, following the procedure in [22]. Specifically, we rescale the pixel value to $[0, 1]$ and randomly rotate the images at angles uniformly sampled from $[-\frac{\pi}{4}, \frac{\pi}{4}]$. The resulting images are used as inputs of the first view. For each image of the first view, we randomly select an image of the same identity from the original dataset, add noise uniformly sampled from $[0, 1]$, and truncate the pixel value to $[0, 1]$. This image is further resized to 24 × 24 to obtain the corresponding image for the second view. As a result, data dimensions of the two views are arbitrarily different. We set aside $10K$ image pairs from the original training set for tuning.

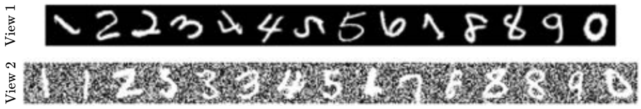
20 Newsgroup (20News).[注1] The dataset contains about $20K$ postings in 20 related categories. We first remove all stop words[注2] from the corpus. We then separately fit them to 110-topic latent Dirichlet allocation (LDA) [23] model and 120-topic probabilistic latent semantic analysis (PLSA) [24] model. As a result, each document is characterized by two topic proportion vectors, whose dimensions are 110 (for the first view) and 120 (for the second view), respectively. We finally split the obtained two-view data with a ratio of 60/20/20 for training/tuning/testing.

NBA-NASCAR sport (NNSpt). This is an image-text dataset collected by [25]. It consists of 420 NBA images and 420 NASCAR images, each of which has an attached short text describing the related content. Each image is normalized to be a 32 × 32 grayscale image; thus, each sample of the first view has a dimension of 1024. The attached short text is preprocessed for the second view, and each text has a 296-dimensional TFIDF [26] feature. We again split the image–text dataset with a ratio of 60/20/20 for training/tuning/testing.

After preprocessing, training parts of the above datasets are given in the following form: $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{l}) = \{(\boldsymbol{x}_i, \boldsymbol{y}_i, l_i) | 1 \leqq i \leqq n, \boldsymbol{x}_i \in \mathbb{R}^{d_x}, \boldsymbol{y}_i \in \mathbb{R}^{d_y}\}$, where l_i is the label of the pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ and n is the number of training samples. To generate two misaligned sequences $\boldsymbol{X}'$ and $\boldsymbol{Y}'$, we use the profile hidden Markov model (pHMM) [27]. There are four types of states in pHMM: i) the `MATCHING` state M_i is a regular state that emits the i^{th} matching sample for $i = 1, ..., n$; ii) the `INSERT` state I_i is intended for emitting sample replication; iii) the `DELETE` state is aimed at the deletion of the matching sample; and iv) two special states B and E indicate the start/end of the sequence.

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
X	M_1	M_2	I_2	M_3	M_4	I_4	M_5	I_5	M_6	M_7	M_8	I_8	M_9	M_{10}	☒
Y	M_1	I_1	M_2	M_3	I_3	I_3	M_4	M_5	M_6	M_7	M_8	I_8	I_8	M_9	M_{10}

(a)



(b)

Figure 3 A toy example of how to generate misaligned sequences. (a) Hidden states generated by pHMM and (b) the corresponding two data sequences generated from the noisy MNIST digits dataset.

Ignoring the `DELETE` state for simplicity, we choose the transition probability as follows: from any state, the next state is `MATCHING` with probability 0.6 and `INSERT` with probability 0.4. While the state M_i at sequence $\boldsymbol{X}'$ corresponds to sample $\boldsymbol{x}_i$, for the state I_i, we replicate $\boldsymbol{x}_i$ by randomly selecting a sample $\boldsymbol{x}_{l=l_i}$ from its class (having the same label). (Similarly for the sequence $\boldsymbol{Y}'$.) Figure 3 shows an example of hidden states generated from pHMM for a given training data: $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{l}) = \{(\boldsymbol{x}_i, \boldsymbol{y}_i, l_i) | 1 \leqq i \leqq 8\}$.

6 2 Compared Methods

We compare DSCA-e, DSCA-s, DSCAE-e and DSCAE-e[注3] with the following baselines:

- Canonical time warping (CTW) [14]—a direct combination of CCA and DTW;
- Canonical soft time warping (CSTW) [28]—a probabilistic extension of CTW, where the alignment is considered a variable that follows Gibbs distribution. The alignment and projection matrices are alternatively optimized using the Expectation–Maximization (EM) algorithm.
- Autoencoder regularized CTW (AECTW) [15]—a variant of CTW with autoencoder-based regularizations;
- Deep CTW (DCTW) [16]—a direct combination of Deep CCA and DTW;
- Locally unsupervised manifold alignment (LUMA) [17]—an unsupervised manifold alignment-based method that establishes a connection between any two samples from the two views by comparing their local geometries;
- Fuzzy granule manifold alignment (FGMA) [19]—a variant of LUMA, where the local geometry information is collected in the fuzzy granule space instead of the original space.

[注1]：The data were retrieved at `http://people.csail.mit.edu/jrennie/20Newsgroups/`.

[注2]：The stop word list can be found at `http://mallet.cs.umass.edu/`.

[注3]：Suffixes -e and -s indicate the versions of the proposed models, where regularization of the employed generalized smooth DTW is entropy and squared l2 norm, respectively.

Dataset	DNNs architectures	
	Mapping (f_x, f_y)	Reconstructing (g_x, g_y)
MNIST	784-1200-1200-1200-d 576-1000-1000-1000-d	1200-1200-1200-784 1000-1000-1000-576
20News	110-500-500-500-d 120-600-600-600-d	500-500-500-110 600-600-600-120
NNSpt	1024-1800-1800-1800-d 296-1200-1200-1200-d	1800-1800-1800-1024 1200-1200-1200-296

Table 1 DNN architectures (number of sigmoid units at each layer) for mapping in DCTW and DSCA. DSCAE is added symmetric DNNs for view reconstruction.

- Generalized unsupervised manifold alignment (GUMA) [18]—another unsupervised manifold alignment-based method that encodes cross-view sample-wise correspondence into a binary matrix that is jointly optimized with the projections;
- Manifold alignment time warping (MATW) [21]—a hybrid method where sample alignment is performed by DTW and where projection matrices are optimized to preserve the underlying structures of the two views;

6 3 Evaluation Measurements

We evaluate these methods by measuring class separation in the learned embedding spaces. Firstly, we perform clustering task on the projections of the first view and evaluate how well the clusters agree with the ground-truth labels. We follows the same procedure as in [22], where spectral clustering [29] is used to handle possibly non-convex cluster shapes. We set the number of clusters to the number of ground-truth classes available in each dataset. Clustering accuracy (ACC) and normalized mutual information (NMI) [30] are used as measurements for assessing the clustering performance. Secondly, we test the accuracy of a simple linear classifier on the learned embeddings. We train one-versus-one linear support vector machines (SVMs) [31] on the projected training set of the first view (label information is used). The trained model is then used to classify projections of the test set, and the percentage of errors is reported as a measurement of classification performance.

6 4 Parameters Tuning

We select the optimal parameters, which return the best evaluation measurement results on the tuning set, for each method. Dimension d of the new subspace is selected from $\{5, 10, 20, 30, 50, 70\}$. For manifold alignment-based methods, the parameter balance between sample matching and geometry preserving are chosen from $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. We found that these methods achieve stable high scores over all the datasets at the parameter value of 0.5 . The trade-off parameters λ for autoencoder regularizations in AECTW and DSCAE are selected using grid search. For the parameters associated with soft whitening regularizations in $\mathcal{L}_{SCA}$, we set $\lambda_x = \lambda_y$, and their values are also determined via grid search. Another important parameter in the proposed methods is β of the generalized smooth DTW, and we select its value from $\{10, 1, 0.1, 0.01, 0.001\}$. We found that our methods work reasonably well at $\beta = 1$. Configurations of DNNs in DCTW, DSCA, and DSCAE are dataset-dependent and summarized in Table 1. Note that in our methods, we add a BN layer with d units, which is not shown in the table, on top of each mapping DNN.

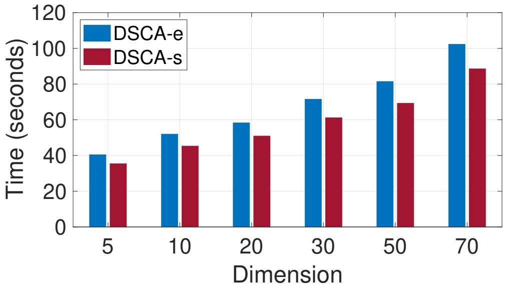


Figure 4 Average times for computing stochastic gradients of DSCA-e and DSCA-s over a batch size ratio $\alpha = 0.1$ (equivalent to a batch size of about $1.5K$) on noisy MNIST digits dataset with different dimensions d of the learned subspace.

6 5 Results and Discussion

Figure 5 visualizes the first view in the original space and its projections in the subspaces learned by different methods. The class separation results are shown in Table 2.

Two-view 20News data consist of topic proportion vectors inferred by LDA (for the first view) and PLSA (for the second view). While LDA has a Dirichlet prior over the topic proportions, PLSA simply considers them as multinomial variables. As a result, LDA encourages a document to focus on a limited number of topics (only some elements of the topic proportions have large magnitudes), while the vectors inferred by PLSA are more spread out. Therefore, the second view can be considered as a noisy version of the first one as in noisy MNIST digits.

Because the noise corrupted geometric information, the manifold alignment-based methods, including LUMA, FGMA, GUMA, and MATW, returned poor results on noisy MNIST and 20News datasets. In contrast, the deep learning-based methods learned the subspaces with much higher class separation results, even in noisy conditions. These methods mapped samples of the same class to similar locations while suppressing noise and/or rotational variation (MNIST) in the data. However, our methods worked much better than DCTW because they implicitly discover sample correspondence (by minimizing the generalized smooth DTW distance between projections of the views) while learning representations. In contrast, DCTW alternates between sample matching (using DTW) and updating the new subspace (using

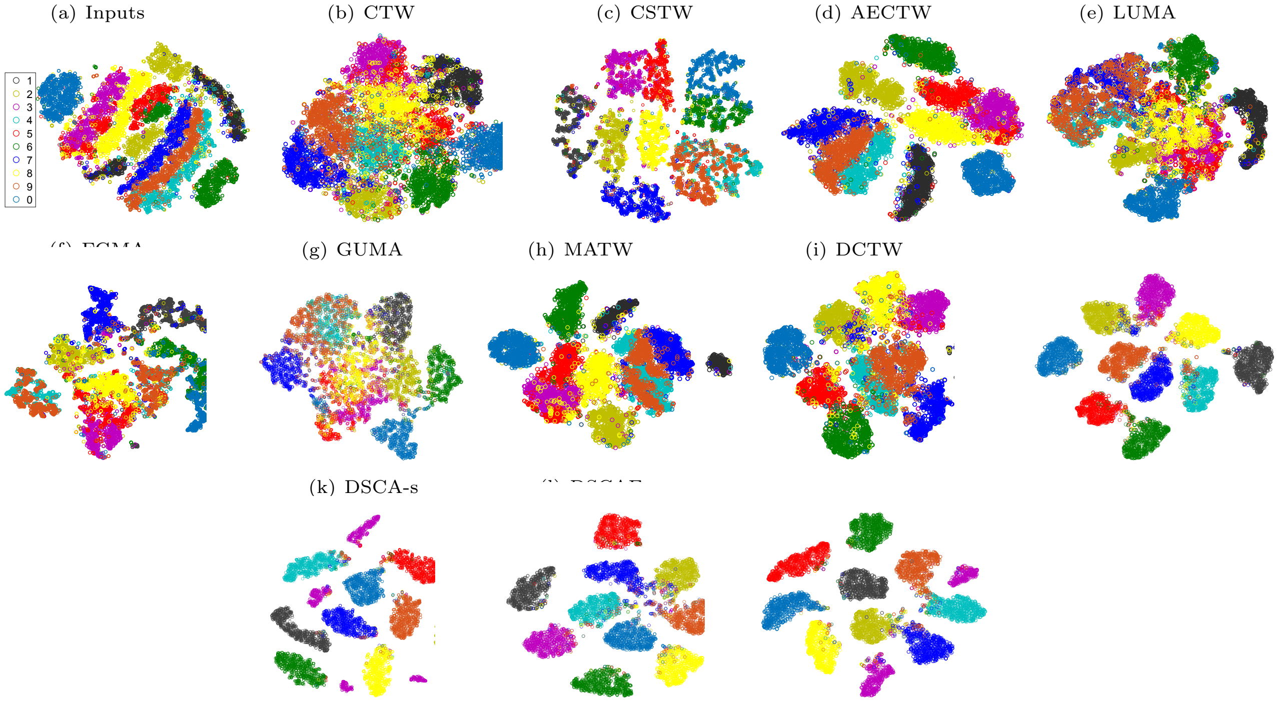


Figure 5 *t*-SNE [32] visualization of the projected test set of noisy MNIST digits on shared subspaces with dimension $d = 10$ returned by different methods.

Method	MNIST			20News			NNspt		
	ACC (%)	NMI (%)	Error (%)	ACC (%)	NMI (%)	Error (%)	ACC (%)	NMI (%)	Error (%)
Inputs	45.14 (5.27)	48.05 (6.77)	15.61 (2.86)	41.47 (4.82)	40.71 (4.41)	30.25 (5.17)	77.49 (2.31)	76.51 (1.98)	10.31 (1.75)
CTW	66.33 (4.47)	52.17 (4.81)	20.28 (3.21)	61.64 (3.27)	60.95 (3.65)	22.33 (4.41)	81.26 (2.11)	82.06 (2.01)	6.29 (1.37)
CSTW	75.16 (1.62)	73.91 (2.01)	7.92 (1.50)	70.04 (2.51)	71.13 (3.00)	16.67 (2.92)	85.11 (1.93)	85.84 (2.02)	4.35 (1.48)
AECTW	82.62 (2.22)	79.66 (2.12)	8.24 (1.72)	79.35 (2.51)	75.17 (2.10)	11.78 (2.97)	91.27 (0.77)	89.32 (0.91)	4.13 (0.71)
LUMA	62.37 (3.21)	61.25 (3.71)	26.23 (3.64)	60.48 (4.12)	57.32 (4.19)	25.52 (4.22)	84.33 (1.54)	83.27 (1.52)	7.32 (1.12)
FGMA	66.51 (2.19)	65.93 (2.87)	24.36 (2.92)	63.57 (3.51)	61.19 (3.42)	21.82 (2.93)	86.46 (1.07)	85.30 (1.15)	6.89 (1.09)
GUMA	60.38 (3.47)	54.25 (2.90)	28.03 (3.14)	59.45 (3.96)	56.11 (3.55)	26.07 (3.18)	80.14 (1.58)	81.26 (1.95)	7.52 (1.46)
MATW	69.18 (4.86)	67.44 (5.13)	21.39 (3.03)	65.23 (3.87)	61.59 (3.64)	23.61 (4.52)	86.64 (1.37)	87.13 (1.34)	6.41 (1.00)
DCTW	86.46 (1.13)	84.22 (2.09)	6.23 (1.46)	83.60 (1.09)	80.22 (1.55)	9.71 (1.61)	95.50 (0.49)	93.33 (0.35)	3.26 (0.34)
DSCA-e	95.12 (0.98)	93.21 (1.34)	**2.81 (0.58)**	88.21 (0.45)	86.76 (1.07)	4.63 (0.54)	98.13 (0.10)	95.64 (0.12)	**0.98 (0.03)**
DSCA-s	90.57 (0.95)	89.93 (1.17)	5.06 (0.92)	87.42 (0.31)	87.82 (0.98)	4.85 (0.66)	97.62 (0.28)	95.37 (0.09)	1.07 (0.12)
DSCAE-e	**96.71 (0.79)**	**95.54 (0.68)**	3.07 (0.96)	**90.18 (0.36)**	**87.94 (0.69)**	**4.20 (0.38)**	**98.65 (0.07)**	**96.01 (0.10)**	1.21 (0.03)
DSCAE-s	91.38 (1.04)	90.19 (1.32)	4.11 (1.14)	89.83 (0.45)	87.35 (1.21)	4.32 (0.81)	98.07 (0.42)	95.59 (0.16)	1.33 (0.08)

Table 2 Performance measures of clustering (ACC, NMI) and classifying (Error) on the projections of the three datasets, using different methods. The data sequences are randomly generated five times using the pHMM-based procedure and the average results along with variances are reported.

DCCA), leading to suboptimal solutions.

The experimental results also show that by carefully tuning the trade-off parameters, combining CTW or DSCA with autoencoders (forming the variants AECTW or DSCAE) can improve their performances. All the methods have better scores on NNSpt because the dataset is less noisy, and our methods again achieved the highest results. We note that each method proposed in this paper has two versions depending on the selection of the regularization $\Omega(\boldsymbol{\eta})$ and their efficiencies are slightly different. Figure 4 shows the average times for computing the stochastic gradients of DSCA-e and DSCA-s over different dimensions d. Because of the sparsity of the gradient induced by squared ℓ_2 norm, training DSCA-s and DSCAE-s are generally faster than DSCA-e and DSCAE-e, respectively. However, this advantage comes at a cost of slightly lower class separation scores as can be observed in Table 2, because $\mathrm{DTW}_{\Omega=\text{squared } \ell_2}$ is a nonexact approximation of the original DTW [7].

7 Conclusion

This paper introduced DSCA, a DNN-based method for multi-view representation learning. Unlike conventional methods that require multi-view data to be equal in size and sample-wise matching, DSCA can implicitly discover sample correspondence while learning representations. In addition, because its objective is unconstrained, we can design an efficient SGD-based algorithm to train DSCA. We also extend our model by adding two DNNs for view reconstructions, forming a new model DSCAE. Through extensive experimentation on different publicly available datasets, our methods were compared with various baselines. The results show that the performances of our methods surpass those of the competitors on all the datasets.

References

[1] Hotelling, Harold. “Relations between two sets of variates.” Biometrika 28, no. 3/4 (1936): 321–377.

[2] Andrew, Galen, Raman Arora, Jeff Bilmes, and Karen Livescu. “Deep canonical correlation analysis.” In International Conference on Machine Learning, pp. 1247–1255. 2013.

[3] Wang, Weiran, Raman Arora, Karen Livescu, and Nathan Srebro. “Stochastic optimization for deep CCA via nonlinear orthogonal iterations.” In Communication, Control, and Computing (Allerton), 2015 53rd Annual Allerton Conference on, pp. 688–695. IEEE, 2015.

[4] Rabiner, Lawrence R., and Biing-Hwang Juang. Fundamentals of Speech Recognition. Vol. 14. Englewood Cliffs: PTR Prentice Hall, 1993.

[5] Ngiam, Jiquan, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. “Multimodal deep learning.” In International Conference on Machine Learning, pp. 689–696. 2011.

[6] Nesterov, Yu. “Smooth minimization of non-smooth functions.” Mathematical Programming 103, no. 1 (2005): 127–152.

[7] Mensch, Arthur, and Mathieu Blondel. “Differentiable dynamic programming for structured prediction and attention.” In 35th International Conference on Machine Learning, vol. 80. 2018.

[8] Danskin, John M. “The theory of max-min, with applications.” SIAM Journal on Applied Mathematics 14, no. 4 (1966): 641-664.

[9] Ioffe and Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift.” In International Conference on Machine Learning, pp. 448–456. 2015.

[10] LeCun, Yann, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. “Backpropagation applied to handwritten zip code recognition.” Neural Computation 1, no. 4 (1989): 541–551.

[11] Nocedal, Jorge, and Stephen J. Wright. Nonlinear Equations. Springer New York, 2006.

[12] Bottou, Lon. “Stochastic gradient learning in neural networks.” Proceedings of Neuro-Nmes 91, no. 8 (1991): 12.

[13] LeCun, Yann, Lon Bottou, Yoshua Bengio, and Patrick Haffner. “Gradient-based learning applied to document recognition.” Proceedings of the IEEE 86, no. 11 (1998): 2278–2324.

[14] Zhou, Feng, and Fernando Torre.“Canonical time warping for alignment of human behavior.” In Advances in Neural Information Processing Systems, pp. 2286–2294. 2009.

[15] Nie, Liquan, Yuanyuan Wang, Xiang Zhang, Xuhui Huang, and Zhigang Luo. “Enhancing temporal alignment with autoencoder regularization.” In Neural Networks (IJCNN), 2016 International Joint Conference on, pp. 4873–4879. IEEE, 2016.

[16] Trigeorgis, George, Mihalis A. Nicolaou, Bjorn W. Schuller, and Stefanos Zafeiriou. “Deep canonical time warping for simultaneous alignment and representation learning of sequences.” IEEE Transactions on Pattern Analysis & Machine Intelligence 5 (2018): 1128–1138.

[17] Wang, Chang, and Sridhar Mahadevan. “Manifold alignment without correspondence.” In Twenty-First International Joint Conference on Artificial Intelligence. 2009.

[18] Cui, Zhen, Hong Chang, Shiguang Shan, and Xilin Chen. “Generalized unsupervised manifold alignment.” In Advances in Neural Information Processing Systems, pp. 2429–2437. 2014.

[19] Li, Wei, Jianwu Xue, Yumin Chen, Xuebai Zhang, Chao Tang, Qiang Zhang, and Yifang Gao. “Fuzzy granule manifold alignment preserving local topology.” IEEE Access (2020).

[20] Gong, Dian, and Gerard Medioni. “Dynamic manifold warping for view invariant action recognition.” In 2011 International Conference on Computer Vision, pp. 571–578. IEEE, 2011.

[21] Vu, Hoa Trong, Clifton Carey, and Sridhar Mahadevan. “Manifold warping: Manifold alignment over time.” In Proceedings of the 26th AAAI Conference on Artificial Intelligence. 2012.

[22] Wang, Weiran, Raman Arora, Karen Livescu, and Jeff Bilmes. “On deep multi-view representation learning.” In International Conference on Machine Learning, pp. 1083–1092. 2015.

[23] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. “Latent Dirichlet allocation.” Journal of Machine Learning Research 3, Jan (2003): 993–1022.

[24] Hofmann, Thomas. “Unsupervised learning by probabilistic latent semantic analysis.” Machine Learning 42, no. 1–2 (2001): 177–196.

[25] Sun, Shiliang. “Multi-view Laplacian support vector machines.” In International Conference on Advanced Data Mining and Applications, pp. 209–222. Springer, Berlin, Heidelberg, 2011.

[26] Salton, Gerard, and Christopher Buckley. “Term-weighting approaches in automatic text retrieval.” Information Processing & Management 24, no. 5 (1988): 513–523.

[27] Eddy, Sean R. “Profile hidden Markov models.” Bioinformatics (Oxford, England) 14, no. 9 (1998): 755–763.

[28] Kawano, Keisuke, Satoshi Koide, and Takuro Kutsuna. “Canonical soft time warping.” In Asian Conference on Machine Learning, pp. 551566. 2019

[29] Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. “On spectral clustering: Analysis and an algorithm.” In Advances in Neural Information Processing Systems, pp. 849–856. 2002.

[30] Cai, Deng, Xiaofei He, and Jiawei Han. “Document clustering using locality preserving indexing.” IEEE Transactions on Knowledge and Data Engineering 17, no. 12 (2005): 1624–1637.

[31] Chang, Chih-Chung, and Chih-Jen Lin. “LIBSVM: A library for support vector machines.” ACM Transactions on Intelligent Systems and Technology (TIST) 2, no. 3 (2011): 27.

[32] Maaten, Laurens van der, and Geoffrey Hinton. “Visualizing data using t-SNE.” Journal of Machine Learning Research 9, Nov (2008): 2579–2605.