Relational Autoencoder for Feature Extraction

Qinxue Meng*, Daniel Catchpoole†, David Skillicorn‡, and Paul J. Kennedy*
* Centre for Artificial Intelligence, Faculty of Engineering and Information Technology,
University of Technology Sydney, Sydney, Australia
† Children's Cancer Research Unit, The Children's Hospital at Westmead, Sydney, Australia
‡ School of Computing, Queen's University at Kingston, Ontario, Canada
Email: Qinxue.Meng@uts.edu.au, Daniel.catchpoole@health.nsw.gov.au, skill@cs.queensu.ca, Paul.Kennedy@uts.edu.au

***Abstract*—Feature extraction becomes increasingly important as data grows high dimensional. Autoencoder as a neural network based feature extraction method achieves great success in generating abstract features of high dimensional data. However, it fails to consider the relationships of data samples which may affect experimental results of using original and new features. In this paper, we propose a Relation Autoencoder model considering both data features and their relationships. We also extend it to work with other major autoencoder models including Sparse Autoencoder, Denoising Autoencoder and Variational Autoencoder. The proposed relational autoencoder models are evaluated on a set of benchmark datasets and the experimental results show that considering data relationships can generate more robust features which achieve lower construction loss and then lower error rate in further classification compared to the other variants of autoencoders.**

I. Introduction

As data becomes increasingly high-dimensional such as genomic information, images, videos and text, reducing dimensionality to generate a high-level representation is considered not only as an important but also necessary data preprocessing step. This is because although machine learning models should, theoretically, be able to work on any number of features, high-dimensional datasets always bring about a series of problems including over-fitting, high-computational complexity and overly complicated models, which gives rise to a well known issue - curse of dimensionality [1]. Another reason for reducing dimensionality is that high-level representations can help people better understand the intrinsic structure of data.

Various methods of dimensionality reduction [2–4] have been proposed and they can be roughly divided into two categories: feature selection and feature extraction. The major difference between them lies in using part or all of input features. For example, considering a given dataset X with a feature set F, feature selection finds a subset of features D_s from all features F ($D_s \subset F$) and the number of selected features is smaller than the original ($|D_s| \ll |F|$) while feature extraction generates a new set of features D_e which are combinations of the original ones F. Generally new features are different from original features ($D_e \nsubseteq F$) and the number of new features, in most cases, is smaller than original features ($|D_e| \ll |F|$). Although feature selection methods, such as Subset Selection [5] and Random Forest [6], are effective in filtering out unnecessary features, if all input features contribute to final results to some extent, feature selection is more likely to give rise to information loss than feature extraction because it fails to use all of them. Therefore, the research focus of dimensionality reduction has been moving towards feature extraction, though not totally.

The initial feature extraction methods [7] are proposed based on projection, mapping input features in the original high-dimensional space to a new low-dimensional space by minimizing information loss. The most famous projection methods are Principal Component Analysis (PCA) [8] and Linear Discriminant Analysis (LDA) [9]. The former one is an unsupervised method, projecting original data into its principal directions by maximizing variance. The latter one is a supervised method to find a linear subspace by optimizing discriminating data from classes. The major drawback of these methods is that they do projection linearly. Subsequent studies [10–14] overcome this issue by employing non-linear kernel functions.

However, projecting from a high-dimensional space to a low-dimensional one is hard to maintain relationships among data samples which gives rise to change the relationship among data samples so as to generate different results of using original and new features. Therefore, recently, exploiting data relationships has been a major focus of dimensionality reduction research. Multidimensional Scaling (MDS) [15] considers the relationship of data samples by transforming distances into similarities for visualization. ISOMAP [16] learns low-dimensional features by retaining the geodesic distances among data samples. Locally Linear Embedding (LLE) [17] preserves data relationships by embedding local neighbourhood when mapping to low-dimensional space. Laplacian Eigenmaps (LE) [18] minimizes the pairwise distances in the projected space by weighting the corresponding distances in the original space. One issue of these methods is that they generally have a fixed or predefined way of capturing local data relationships in the high-dimensional space which may not be accurate and valid in a low-dimensional space. Another issue is that the major work of these studies maps data from high-dimensional space to low-dimensional space by extracting features once instead of stacking them to gradually generate deeper levels of representation.

Autoencoders [19] use artificial neural networks to reduce dimensionality by minimizing reconstruction loss. Thus, it is easy to stack by adding hidden layers. Because of this, au-

978-1-5090-6182-2/17/$31.00 ©2017 IEEE

toencoders and extensions such as Sparse Autoencoders [20], Denoising Autoencoders [21], Contractive Autoencoders [22] and Variational Autoencoders [23], demonstrate a promising ability to extract meaningful features, especially in image processing and natural language processing. Yet, these methods only consider data reconstruction and ignore to explicitly model its relationship. A recent study [24] proposes a Generalized Autoencoder (GAE) which focuses on modifying autoencoder to consider data relationship by adding a weighted relational function. Specifically, it minimizes the weighted distances between reconstructed instances and the original ones. Although subsequent applications [25–29] confirm that considering data relationship can improve the performance of autoencoder in dimensionality reduction, the Generalized Autoencoder model has some drawbacks. Firstly, determining the weights for each pair-wise distance is very challenging and this practice is risky in converting GAE into a supervised model when some relationshps are over emphasized and others are neglected. Meanwhile focusing on minimizing the loss of data relationships but ignoring the loss of data itself is very likely to lose information. For example, some data attributes contribute more to data relationships than the other. Focusing on reconstructing data relationships may emphasize part of attributes and ignores the others.

To deal with above issues, we proposed a Relation Autoencoder (RAE) for dimensionality reduction and the contributions are summarized as

1) RAE considers both data features and their relationships by minimizing the loss of them.
2) To alleviate the increased computational complex of considering data relationships, the proposed Relational Autoencoder model can filter out weak and trivial relationships by adding activation functions.
3) The relationship model has been extended to work with other baseline autoencoder models including Sparse Autoencoder (SAE), Denoising Autoencoder (DAE) and Variational Autoencoder (VAE).

In this paper, we comprehensively evaluate the proposed Relational Autoencoder on a set of benchmark datasets. Specifically, in the experiment, we firstly compare the performance of our proposed model with another recently published relationship-based autoencoder model (GAE). Then we compare the performance of these baseline autoencoder models with their extended versions.

The rest of the paper starts from reviewing related work of autoencoders in Section II followed by problem definition and basic autoencoders in Section III. Section IV presents the proposed Relational Autoencoder model and its extensions of the other baseline autoencoder models. The experimental datasets and results are covered in Section V and Section VI respectively. Finally, we discuss the proposed method and conclude the paper in Section VII.

II. Literature review

Autoencoder was initially introduced in the later 1980s [30] as a linear feature extraction method. Unlike latent space approaches which map data into a high dimensional space, autoencoder aims to learn a simpler representation of data by mapping the original data into a low-dimensional space. The main principle of autoencoder follows from the name. "Auto" presents that this method is unsupervised and "encoder" means it learns another representation of data. Specifically, autoencoder learns a encoded representation by minimizing the loss between the original data and the data decoded from this representation. In 1989, Baldi and Hornik [31] investigated the relationship between a one-layer autoencoder and principal component analysis (PCA). They found that the new compressed features learnt by linear autoencoder is similar to principal components. However, the computational complexity of training an autoencoder is much higher than PCA because the major computational complexity of PCA is based on matrix decomposition. This limits the application of linear autoencoders.

Later, with the involvement of non-linear activation functions, autoencoder becomes non-linear and is capable of learning more useful features [32] than linear feature extraction methods. Non-linear autoencoders are not advantaged than the other non-linear feature extraction methods as it takes long time to train them.

The recent revival of interest in autoencoders is due to the success of effectively training deep architectures because traditional gradient-based optimization strategies are not effective when hidden layers are stacked multiple times with non-linearities. In 2006, Hinton [33] empirically trained a deep network architecture by sequentially and greedily optimizing weights of each layer in a Restricted Boltzmann Machine (RBM) model to achieve global gradient-based optimization. Bengio [34] achieved success in training a stacked autoencoder on the MNIST dataset with a similar greedy layerwise approach. This training approach overcomes the problematic non-convex optimization which prevents deep network structure. Subsequent studies show that stacked autoencoder model can learn meaningful, abstract features and thus achieve better classification results in high-dimensional data, such as images and texts [35–38]. The training efficiency of stacked autoencoder is further improved by changing layer weight initialization [39].

As the training efficiency improves, autoencoder becomes increasingly popular. Soon it was found that as layers are stacked, the weights of deeper layers increase sharply because of matrix multiplication and then the importances of these weights become larger than the initial input features. This overfitting issue gives rise to the fact that representations of deep layers are more likely dependent on the network structure instead of the initial input features. Poultney et al. [40] presented an idea to increase the sparsity of network structure so as to limit the increasing of weights. Goodfellow et al. [41] added a regularization term in the loss function of autoencoder to impose a penalty on large weights. Vincent et al. [42] proposed a Denoising Autoencoder (DAE) to solve this issue by adding noises in the input. The proposed DAE model aims not to reconstruct the original input but to reconstruct

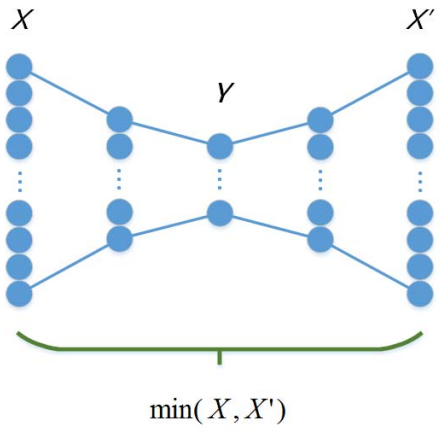


Fig. 1. Traditional autoencoder generates new feature set Y by minimizing the reconstruction loss between X and X'.

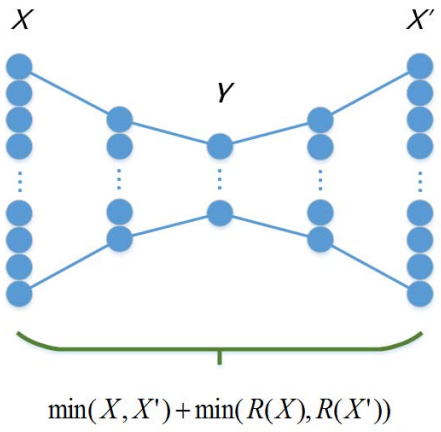


Fig. 2. Relational autoencoder generates new feature set Y by minimizing the loss of reconstructing X and maintaining relationships among X.

a corrupted input which is typically corrupted by adding Gaussian noise. The previous studies have no control of the distribution of hidden-layer representations. So Variational Autoencoder (VAE) [23] is proposed to generate desired distributions of representations in hidden layers.

Autoencoders are an unsupervised dimension reduction process and the reduced high-level features generally contain the major information of original data. This makes autoencoders not sensitive to slight variations. To make them sensitive to slight variations, Rifai et al. [22] proposed a Contractive Autoencoder (CAE). In fact, maintaining mutual relationships among data samples works as an effective way of making autoencoders sensitive to slight variations. Because of this, Wang et al. [24] proposed a Generalized Autoencoder (GAE) targeting at reconstructing the data relationships instead of the data features. A series of applications [25–29] of GAE confirm that maintaining data relationships can achieve better results but results are highly dependent on how to define and choose distance weights. The practice of having sophisticated, pre-defined distance weights is risky in converting GAE into a supervised model because of assigning high weights to selected relationships is very subjective and may be biased. As a result, we propose a Relation Autoencoder (RAE) for dimensionality reduction by minimising both the loss of data features and relationships.

III. Preliminaries

This section begins with a definition of feature extraction followed by a description of basic autoencoder models before presenting our proposed Relational Autoencoder (RAE) model.

A. Feature Extraction

Feature extraction transforms data from the original, high-dimensional space to a relatively low-dimensional space. This transformation can be linear or nonlinear. Specifically, considering a given dataset X with n samples and m features, the original feature set is denoted as F_O and a feature extraction function T generates new feature set F_N where $|F_N| < |F_O|$. Generally, the objective functions of feature extraction methods minimize the difference between the original space $F_O(X)$ and the new space $F_N(X)$ and changing the objective functions can convert feature extraction from unsupervised methods to supervised methods such as Linear Discriminant Analysis (LDA).

B. Basic Autoencoder

Simply, an autoencoder (AE) is composed of two parts, an encoder and a decoder. Considering a data sample X with n samples and m features, the output of encoder Y represents the reduced representation of X and the decoder is tuned to reconstruct the original dataset X from the encoder's representation Y by minimizing the difference between X and X' as illustrated in Fig. 1. Specifically, the encoder is a function f that maps an input X to hidden representation Y. The process is formulated as

$$Y = f(X) = s_f(WX + b_X) \tag{1}$$

where s_f is a nonlinear activation function and if it is an identity function, the autoencoder will do linear projection. The encoder is parameterized by a weight matrix W and a bias vector $b \in R^n$.

The decoder function g maps hidden representation Y back to a reconstruction X':

$$X' = g(Y) = s_g(W'Y + b_Y) \tag{2}$$

where s_g is the decoder's activation function, typically either the identity (yielding linear reconstruction) or a sigmoid. The decoder's parameters are a bias vector b_y and matrix W'. In this paper we only explore the tied weights case where $W' = W^T$.

Training an autoencoder involves finding parameters $\theta = (W, b_X, b_Y)$ that minimize the reconstruction loss on the given dataset X and the objective function is given as

$$\Theta = \min_{\theta} L(X, X') = \min_{\theta} L(X, g(f(X))) \tag{3}$$

For linear reconstruction, the reconstruction loss (L_1) is generally from the squared error:

$$L_1(\theta) = \sum_{i=1}^{n} ||x_i - x_i'||^2 = \sum_{i=1}^{n} ||x_i - g(f(x_i))||^2 \quad (4)$$

For nonlinear reconstruction, the reconstruction loss (L_2) is generally from cross-entropy:

$$L_2(\theta) = -\sum_{i=1}^{n} [x_i \log(y_i) + (1 - x_i) \log(1 - y_i)] \quad (5)$$

where $x_i \in X$, $x_i' \in X'$ and $y_i \in Y$.

C. Stacked Autoencoder

As a neural network model based feature extraction method, a major advantage of Autoencoder is that it is easy to stack for generating different levels of new features to represent original ones by adding hidden layers. For an l-layer stacked autoencoder, the process of encoding is

$$Y = f_l(\dots f_i(\dots f_1(X))) \quad (6)$$

where f_i is the encoding function of layer i. The corresponding decoding function is

$$X' = g_l(\dots g_i(\dots g_1(Y))) \quad (7)$$

where g_i is the decoding function of layer i and the Stacked Autoencoder can be trained by greedy layerwise feed-forward approach.

IV. Relational Autoencoder (RAE)

The traditional autoencoder generates new features by minimizing the reconstruction loss of the data. This motivates us to propose a Relational Autoencoder (RAE) to minimize the reconstruction loss of both data and their relationships. The objective function of RAE is defined as

$$\Theta = (1 - \alpha) \min_{\theta} L(X, X') + \alpha \min_{\theta} L(R(X), R(X')) \quad (8)$$

where $R(X)$ represents the relationship among data samples in X and $R(X')$ represents the relationship among data samples in X' as illustrated in Fig. 2. Parameter α is a scale parameter to control the weights of the data reconstruction loss and the relationship reconstruction loss and θ is the neural network parameter of the autoencoder.

Data relationship can be modelled in multiple ways. In this paper, we present data relationship by their similarities where $R(X)$ is the multiplication of X and X^T. Then the objective function is

$$\Theta = (1 - \alpha) \min_{\theta} L(X, X') + \alpha \min_{\theta} L(XX^T, X'X'^T) \quad (9)$$

In order to improve computational efficiency and filter out unnecessary relationships, we use activation functions to control weights of similarities. In this paper, we use the rectifier function [43] to achieve this.

$$\tau_t(r_{ij}) = \begin{cases} r_{ij}, & \text{if } r_{ij} \geq t, \\ 0, & \text{otherwise}, \end{cases} \quad (10)$$

Algorithm 1 Iterative learning procedure of RAE

```
function RAE(X, t, λ, N_l, s_f, ε)
    L = Equ. 11;
    loss_p = 0;
    loss_c = 0;
    Model = NN();    ▷ initialize a neural network model
    for i = 1 to |N| do
        Model.addLayer(n_i, s_f)
    end for
    while True do
        loss_c = Model.train(L, SGD);
        if (loss_c - loss_p ⩽ ε) then
            Break;
        else
            loss_p = loss_c;
        end if
    end while
return Model
end function
```

where t is a threshold to filter out weak and trivial relationships. Then the objective function of RAE is defined as

$$\begin{aligned} \Theta = (1 - \alpha) \min_{\theta} L(X, X') \\ + \alpha \min_{\theta} L(\tau_t(XX^T), \tau_t(X'X'^T)) \end{aligned} \quad (11)$$

In this paper, we choose the loss function L as squared error.

The pseudo-code of the proposed Relational Autoencoder (RAE) is described in Algorithm 1 where the input parameters are the input dataset (X), parameter of the rectifier function (t), regularization weight (λ), the number of hidden neurons in layers (N), activation function (s_f) and a threshold (ε) to determine whether the loss has converged. Among them, N is a vector and n_i is the number of neurons of ith layer where i is from 1 to $|N|$. The proposed RAE model starts with defining a loss function (L). Then it initializes a neural network model and iteratively add layers into the network model based on N and s_f. The network is trained by stochastic gradient descent (SGD) and updates θ in the loss function (L) until the difference between current-loop loss (loss_c) and the previous-loop loss (loss_p) is smaller than a predefined threshold (ε) which means it has converged.

A. Extension to Sparse Autoencoder (SAE)

The objective function of autoencoder reconstructs the input. During the training process, the weights of hidden neurons are the combination of previous layers and these weights increase as layers get deep. High weights of hidden layers make the generated features more dependent on the network structure rather than the input. In order to avoid this, Sparse Autoencoder (SAE) imposes weight-decay regularization so as to keep neuron weights small. The objective function of SAE is

$$\Theta = \alpha \min_{\theta} L(X, X') + \lambda ||W||^2 \quad (12)$$

where $||W||^2$ is a weight-decay regularization term to guarantee weight matrix W having small elements. Parameter λ is a hyper-parameter to control the strength of the regularization. We extend Sparse Autoencoder (SAE) to a Relational Sparse Autoencoder (RSAE) model by considering the data relationships. The objective function is defined as

$$\begin{aligned}\Theta = (1-\alpha)\min_{\theta} L(X, X') \\ + \alpha \min_{\theta} L(\tau_t(XX^T), \tau_t(X'X'^T)) \\ + \lambda||W||^2\end{aligned} \tag{13}$$

B. Extension to Denoising Autoencoder (DAE)

Denoising Autoencoder (DAE) is proposed to generate better features by imposing an alternative form of regularization. The main principle of DAE is to corrupt a part of the input features of a given dataset X before sending it to an autoencoder model, train the network to reconstruct the destroyed input $\tilde{X}$ and then minimize the loss between the reconstructed $\tilde{X}'$ and original X. The objective function of DAE is

$$\Theta = \min_{\theta}[L(X, g(f(\tilde{X})))] \text{ s.t. } \tilde{X} \sim q(\tilde{X}|X) \tag{14}$$

where $\tilde{X}$ is corrupted X from a stochastic corruption process $q(\tilde{X}|X)$ and the objective function is optimized by stochastic gradient descent. Here we extend the proposed Relation Autoencoder model to a Relational Denoising Autoencoder (RDAE) model by considering data relationships. The model minimizes the loss between data X and corrupted data $\tilde{X}$ and the loss between data relationship XX^T and corrupted data relationship $\tilde{X}\tilde{X}^T$. The objective function is defined as

$$\begin{aligned}\Theta = (1-\alpha)\min_{\theta} L(X, g(f(\tilde{X})) \\ + \alpha \min_{\theta} L(\tau_t(XX^T), \tau_t(\tilde{X}\tilde{X}^T)) \\ \text{s. t. } \tilde{X} \sim q(\tilde{X}|X)\end{aligned} \tag{15}$$

In this paper, we consider corruptions as additive isotropic Gaussian noise: $\tilde{X} = X + \Delta$ where $\tilde{X} \sim N(0, \delta^2)$ and δ is the standard deviation of X.

C. Extension to Variational Autoencoder (VAE)

Variational Autoencoder (VAE) is a different to the other types of autoencoder model. It makes a strong assumption concerning the distribution of latent neurons and tries to minimize the difference between a posterior distribution and the distribution of latent neurons with difference measured by the Kullback-Leibler divergence [44]. The objective function of VAE is

$$\Theta = \min D_{KL}(q_\phi(Y|X)||p_\theta(X|Y)) \tag{16}$$

where $q_\phi(Y|X)$ is the encoding process to calculate the probability of Y based on the input X while $p_\theta(X|Y)$ is the decoding process to reconstruct X. Generally Y is a predefined Gaussian distribution, such as $N(0, 1)$. Therefore the extended Relational Variational Autoencoder (RVAE) is defined as

$$\begin{aligned}\Theta = (1-\alpha)\min D_{KL}(q_\phi(Y|X)||p_\theta(X|Y)) \\ + \alpha \min D_{KL}(q_\phi(Y|XX^T)||p_\theta(XX^T|Y))\end{aligned} \tag{17}$$

V. Datasets

The datasets used in this paper to evaluate the proposed Relational Autoencoder are two image datasets, MNIST[1] and CIFAR-10[2]. The MNIST dataset is a well known database of handwritten digits which contains a training set of 60,000 examples and a test set of 10,000 samples. The CIFAR-10 dataset contains 60,000 images which are labelled with 10 classes with each class having 6,000 images. The training set of CIFAR-10 has 50,000 samples while the test set has 10,000 samples.

VI. Experiment

The experiment firstly compares the performance of the proposed Relational Autoencoder (RAE) model against basic autoencoder (BAE) and Generative Autoencoder (GAE) in reconstruction loss and classification accuracy. Then we compare the performance of the extended Relational Sparse Autoencoder (RSAE), Relational Denoising Autoencoder (RDAE) and Relational Variational Autoencoder (RVAE) with their corresponding original versions to estimate the effects of considering relationship in the depth.

A. Experiment Setting

All autoencoder models were tested with the same configuration on the same dataset. Specifically, we use tied weights ($W' = W^T$) in building network structure. The activation function of each layer is sigmoid for both encoder and decoder. The optimization functions of reconstruction loss are listed and described in Section IV and they are trained by stochastic gradient descent (SGD) for 400 epochs. The number of neurons of each layer is determined as

$$n_{i+1} = \begin{cases}\log(n_i), & \text{if } n_{i+1} \geq l_t, \\ l_t, & \text{otherwise,}\end{cases} \tag{18}$$

where n_i is the number of neurons in layer i and n_{i+1} is the number of neurons in layer $i+1$. As the network structure goes deeper, the number of neurons gradually decreases to a predefined threshold l_t. To increase the training efficiency, we use Xavier [45] method to initialize layer weights

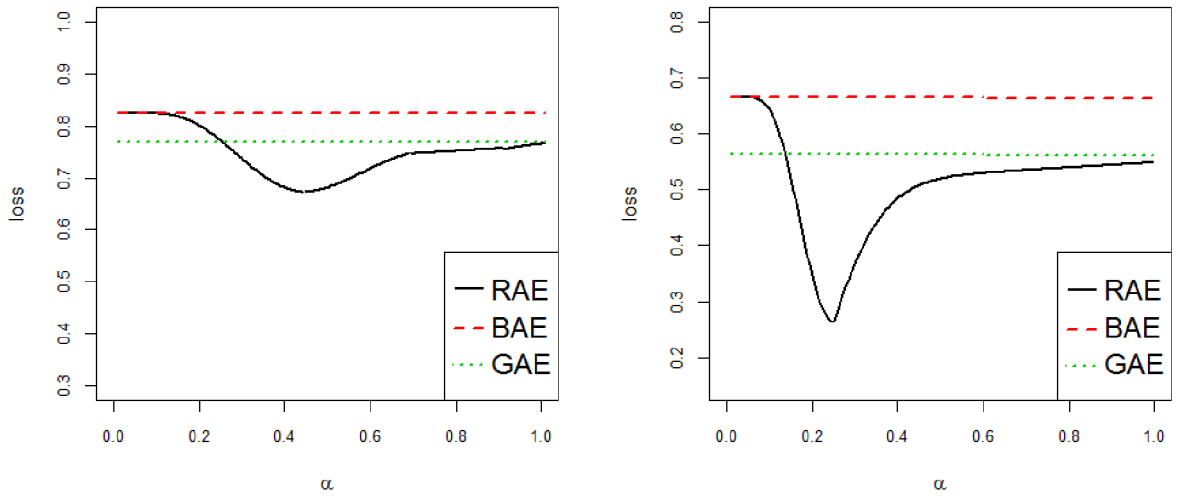
$$w_{ij} = U\left[-\frac{1}{\sqrt{n_{i-1}}}, \frac{1}{\sqrt{n_{i-1}}}\right] \tag{19}$$

where w_{ij} is weight of jth neuron in layer i and $U[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$ is the uniform distribution in the interval $(-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}])$ and n_{i-1} is the number of neurons in the previous layer.

All autoencoders are trained based on their loss function respectively. But in order to compare the performance of them in feature extraction, the reconstruction loss is measured by

[1]http://yann.lecun.com/exdb/mnist/

[2]https://www.cs.toronto.edu/ kriz/cifar.html



(a) MNIST (b) CIFAR-10

Fig. 3. The comparative results of reconstruction loss on the MNIST and CIFAR-10 dataset among Relational Autoencoder (RAE), basic autoencoder (BAE) and Generative Autoencoder (GAE). It describes how information loss changes as the scale parameter α changes.

Mean Squared Error (MSE). Classification is done by softmax regression based on the extracted features from autoencoder models. To estimate the generation of models, we use 10-fold cross validation in both unsupervised feature extraction and classification.

B. Comparing to BAE and GAE

We firstly compare the performance of our proposed Relational Autoencoder (RAE) with basic autoencoder (BAE) and Generative Autoencoder (GAE) in terms of reconstruction on MNIST and CIFAR-10. For GAE, we set similarity to be the weight of each pairwise relationship. In the experiment, we explore different values of scaling parameter α which determines the weights of reconstructing data and relationship (Equation 11). The value of α ranges from 0 to 1 in step of 0.02. Meanwhile, because BAE and GAE has no such parameter, their reconstruction loss is not changed as α changes and the results are illustrated in Fig. 3.

We observe that generally the reconstruction loss of GAE is less than BAE which confirms that considering data relationship is able to reduce information loss in the process of encoding and decoding. The performance of the proposed RAE autoenocder model changes as the scaling parameter α changes. Generally, it starts to generate similar results with BAE because α is small focusing on reconstructing data rather than relationships. As α increases, the performance of GAE continuously decreases until to a tough. Then as α keeps increasing, the information loss increases as well because the model begins to over-emphasize relationship reconstruction. It is interesting to see that even if the proposed RAE model considers relationship only, the performance of RAE is better than GAE as RAE uses the activation function to filter out weak relationships. Thus the performance of the proposed RAE autoencoder model is determined by scaling parameter α and the value of α depends on the dataset.

TABLE I
EXPERIMENTAL RESULTS OF AUTOENCODER MODELS

Model	MNIST		CIFAR-10	
	Loss	Error	Loss	Error
RAE	0.677	3.8%	0.281	12.7%
BAE	0.813	8.9%	0.682	15.6%
GAE	0.782	5.7%	0.574	14.9%
RSAE	0.296	1.8%	0.292	13.4%
SAE	0.312	2.2%	0.331	14.2%
RDAE	0.217	1.1%	0.216	10.5%
DAE	0.269	1.6%	0.229	11.7%
RVAE	0.183	0.9%	0.417	17.3%
VAE	0.201	1.2%	0.552	21.2%

Another interesting finding is that the proposed RAE model achieves better results in CIFAR-10 than MNIST. This may be because the CIFAR-10 contains more complex images than MNIST and for complex datasets, maintaining data relationship is of much importance. For classification, RAE achieves the lowest error rate (3.8%) followed by GAE (5.7%) and BAE (8.9%).

C. Comparing extend autoencoder variations to original ones

In this experiment, we compare the performance of the extended variants of autoencoders to their original versions and detailed results are listed in Table I. We observe that considering data relationships contributes to decreasing reconstruction loss suggesting that autoencoders can generate more robust and meaningful features with less information loss and these features are the key to achieve good classification results.

VII. Conclusion

In this paper, we propose a Relation Autoencoder model which can extract high-level features based on both data itself and their relationships. We extend this principle to other major autoencoder models including Sparse Autoencoder, Denoising Autoencoder and Variational Autoencoder so as to enable them consider data relationships as well. The proposed relational autoencoder models are evaluated on MNIST and CIFAR-10 datasets and the experimental results show that considering data relationships can decrease reconstruction loss and therefore generate more robust features. Better features contribute to improved classification results.

References

[1] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of Computing*. ACM, 1998, pp. 604–613.

[2] F. Wang and J. Sun, "Survey on distance metric learning and dimensionality reduction in data mining," *Data Mining and Knowledge Discovery*, vol. 29, no. 2, pp. 534–564, 2015.

[3] J. P. Cunningham and Z. Ghahramani, "Linear dimensionality reduction: Survey, insights, and generalizations," *Journal of Machine Learning Research*, vol. 16, pp. 2859–2900, 2015.

[4] N. Akkarapatty, A. Muralidharan, N. S. Raj, and P. Vinod, "Dimensionality reduction techniques for text mining," *Collaborative Filtering Using Data Mining and Analysis*, p. 49, 2016.

[5] G. H. John, R. Kohavi, K. Pfleger *et al.*, "Irrelevant features and the subset selection problem," in *Machine learning: proceedings of the eleventh international conference*, 1994, pp. 121–129.

[6] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[7] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Science and Information Conference (SAI), 2014*. IEEE, 2014, pp. 372–378.

[8] U. Demšar, P. Harris, C. Brunsdon, A. S. Fotheringham, and S. McLoone, "Principal component analysis on spatial data: an overview," *Annals of the Association of American Geographers*, vol. 103, no. 1, pp. 106–128, 2013.

[9] A. Sharma and K. K. Paliwal, "Linear discriminant analysis for the small sample size problem: an overview," *International Journal of Machine Learning and Cybernetics*, vol. 6, no. 3, pp. 443–454, 2015.

[10] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International Conference on Artificial Neural Networks*. Springer, 1997, pp. 583–588.

[11] J.-M. Lee, C. Yoo, S. W. Choi, P. A. Vanrolleghem, and I.-B. Lee, "Nonlinear process monitoring using kernel principal component analysis," *Chemical Engineering Science*, vol. 59, no. 1, pp. 223–234, 2004.

[12] P. Honeine, "Online kernel principal component analysis: A reduced-order model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1814–1826, 2012.

[13] L. Lebart, "Correspondence analysis," in *Data Science, Classification, and Related Methods: Proceedings of the Fifth Conference of the International Federation of Classification Societies (IFCS-96), Kobe, Japan, March 27–30, 1996*. Springer Science & Business Media, 2013, p. 423.

[14] D. Lopez-Paz, S. Sra, A. J. Smola, Z. Ghahramani, and B. Schölkopf, "Randomized nonlinear component analysis." in *ICML*, 2014, pp. 1359–1367.

[15] F. W. Young, *Multidimensional scaling: History, theory, and applications*. Psychology Press, 2013.

[16] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[17] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[18] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[19] D. E. Rumelhart and J. L. McClelland, *Learning Internal Representations by Error Propagation*. MIT Press, 1986, pp. 318–362.

[20] J. Deng, Z. Zhang, E. Marchi, and B. Schuller, "Sparse autoencoder-based feature transfer learning for speech emotion recognition," in *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*. IEEE, 2013, pp. 511–516.

[21] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[22] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 833–840.

[23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural*

Information Processing Systems, 2014, pp. 2672–2680.

[24] W. Wang, Y. Huang, Y. Wang, and L. Wang, "Generalized autoencoder: a neural network framework for dimensionality reduction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 490–497.

[25] Z. Camlica, H. Tizhoosh, and F. Khalvati, "Autoencoding the retrieval relevance of medical images," in *Image Processing Theory, Tools and Applications (IPTA), 2015 International Conference on*. IEEE, 2015, pp. 550–555.

[26] S. Stober, A. Sternin, A. M. Owen, and J. A. Grahn, "Deep feature learning for eeg recordings," *arXiv preprint arXiv:1511.04306*, 2015.

[27] L. Gao, J. Song, X. Liu, J. Shao, J. Liu, and J. Shao, "Learning in high-dimensional multimedia data: the state of the art," *Multimedia Systems*, pp. 1–11, 2015.

[28] Y. Wang, H. Yao, S. Zhao, and Y. Zheng, "Dimensionality reduction strategy based on auto-encoder," in *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service*. ACM, 2015, p. 63.

[29] L. Meng, S. Ding, and Y. Xue, "Research on denoising sparse autoencoder," *International Journal of Machine Learning and Cybernetics*, pp. 1–11, 2016.

[30] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagation errors," *Nature*, vol. 323, pp. 533–536, 1986.

[31] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural networks*, vol. 2, no. 1, pp. 53–58, 1989.

[32] N. Japkowicz, S. J. Hanson, and M. A. Gluck, "Nonlinear autoassociation is not equivalent to pca," *Neural computation*, vol. 12, no. 3, pp. 531–545, 2000.

[33] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[34] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proceedings of the 24th international conference on Machine Learning*. ACM, 2007, pp. 473–480.

[35] K. Jarrett, K. Kavukcuoglu, Y. Lecun *et al.*, "What is the best multi-stage architecture for object recognition?" in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 2146–2153.

[36] J.-L. Vincent, J. Rello, J. Marshall, E. Silva, A. Anzueto, C. D. Martin, R. Moreno, J. Lipman, C. Gomersall, Y. Sakr *et al.*, "International study of the prevalence and outcomes of infection in intensive care units," *JAMA*, vol. 302, no. 21, pp. 2323–2329, 2009.

[37] W. W. Ng, G. Zeng, J. Zhang, D. S. Yeung, and W. Pedrycz, "Dual autoencoders features for imbalance classification problem," *Pattern Recognition*, vol. 60, pp. 875–889, 2016.

[38] H.-C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach, "Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1930–1943, 2013.

[39] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *Journal of Machine Learning Research*, vol. 11, no. 2 Feb, pp. 625–660, 2010.

[40] C. Poultney, S. Chopra, Y. L. Cun *et al.*, "Efficient learning of sparse representations with an energy-based model," in *Advances in neural information processing systems*, 2006, pp. 1137–1144.

[41] I. Goodfellow, H. Lee, Q. V. Le, A. Saxe, and A. Y. Ng, "Measuring invariances in deep networks," in *Advances in Neural Information Processing Systems*, 2009, pp. 646–654.

[42] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[44] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[45] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." in *Aistats*, vol. 9, 2010, pp. 249–256.