IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 11, 2021, accepted February 23, 2021, date of publication March 9, 2021, date of current version March 17, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3064819

ResNet Autoencoders for Unsupervised Feature Learning From High-Dimensional Data: Deep Models Resistant to Performance Degradation

CHATHURIKA S. WICKRAMASINGHE, DANIEL L. MARINO, AND MILOS MANIC, (Fellow, IEEE)
Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23220, USA
Corresponding author: Chathurika S. Wickramasinghe (brahmanacsw@vcu.edu)

ABSTRACT Efficient modeling of high-dimensional data requires extracting only relevant dimensions through feature learning. Unsupervised feature learning has gained tremendous attention due to its unbiased approach, no need for prior knowledge or expensive manual processing, and ability to handle exponential data growth. Deep Autoencoder (AE) is a state-of-the-art deep neural network for unsupervised feature learning, which learns embedded-representations using a series of stacked layers. However, as the AE network gets deeper, these learned embedded-representations can deteriorate due to vanishing gradient, leading to performance degradation. This article presents ResNet Autoencoder (RAE) and its convolutional version (C-RAE) for unsupervised feature learning. The advantage of RAE and C-RAE is that it enables the user to add residual connections for increased network capacity without incurring the cost of degradation for unsupervised feature learning compared to standard AEs. While RAE and C-RAE inherit all the advantages of AEs, such as automated non-linear feature extraction and unsupervised learning, they also allow users to design larger networks without adverse effects on feature learning performance. We performed classification on learned embedded-representation to evaluate RAE and C-RAE. RAE and C-RAE were compared against AEs on MNIST, Fashion MNIST, and CIFAR10 datasets. When increasing the number of layers, C-RAE outperformed AE by showing significantly lower performance degradation of classification accuracy (less than 3%) compared to AE (33% to 65%). Further, C-RAE exhibited higher mean accuracy and lower variance of accuracy than standard AE. When comparing RAE and C-RAE with widely used feature learning methods (Convolutional AE, PCA, ICA, LLE, Factor Analysis, and SVD), C-RAE showed the highest accuracy.

INDEX TERMS Deep learning, unsupervised learning, autoencoders, ResNet, classification, deep embedded classification, feature learning, dimension reduction.

I. INTRODUCTION

In this era of industrial big data, a massive amount of data is available to the public through various industries such as intelligent transportation [1], [2], power grids [3], cloud computing [4], and finance [5]. Knowledge extraction on these data is crucial for continuous improvements, process automation, and resilience improvements of these industrial systems [6]. Even though data availability increases exponentially with time, these multi-variety data has many intricacies such as incompleteness, high-dimensionality, noise, and rarely labeled [7]. This article focuses on two main intricacies; high-dimensional and unlabeled data.

The first area of focus is the high-dimensionality of data. The reliability of knowledge extraction methods generally deteriorates due to the curse of dimensionality [8]. In other words, extracting relevant features leads to a reduced number of features that results in efficient knowledge extraction methods with high accuracy [9]. Therefore, when using high-dimensional data for data-driven machine learning tasks, it is necessary to capture only the relevant information [2], [8], [10]. Extraction of relevant features and reduction of input data dimensions are performed using various feature learning and dimensionality reduction techniques. This is achieved by performing non-linear mapping of

The associate editor coordinating the review of this manuscript and approving it for publication was Hao Luo.

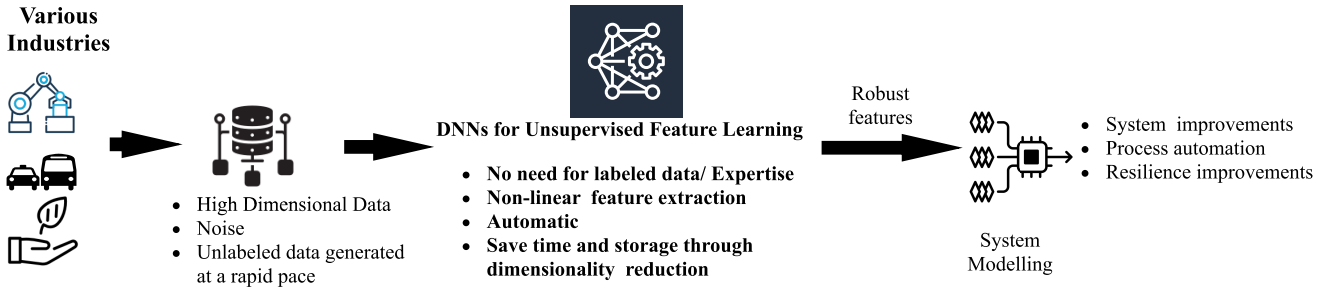


FIGURE 1. The need for Deep Neural Networks (DNN) based unsupervised feature learning and its advantages.

input data into an embedded representation [11]–[13]. Since the embedded representation only contains relevant information, we can use these learned embedded representations to perform various machine learning tasks with improved reliability.

The second area of focus is the abundance of unlabeled data. Real-world settings bring the challenge of dealing with high volumes of unlabeled data. The manual labeling process is time-consuming, expensive, and requires the expertise of the data [14]. Further, supervised feature learning not only is unable to take advantage of unlabelled data, but it also can result in biases by relying on labeled data. Therefore, unsupervised deep learning based feature learning(feature extraction) has gained tremendous attention.

Many dimensionality reduction based unsupervised feature learning methods has been proposed to address the above two problems. Widely used unsupervised feature learning techniques include Principle component analysis (PCA) [15], Independent component analysis (ICA), Locally Linear Embedding (LLE) [15], Factor Analysis embedding, and SVD embedding. Recently, Deep Learning has shown remarkable performance in many areas. It has been successfully used to convert high-dimensional feature spaces into new embedded representations with relevant and robust features [8], [14], [16]. This effective transformation of the input data space to embedded space has been achieved through unsupervised deep learning methods such as deep convolutional autoencoders (C-AEs) [11], [13]. Figure 1 shows current applications of Deep Neural Network (DNN) based approaches for various industrial applications such as process automation and resilience improvement.

Even though Deep learning had become the primary technique with state-of-the-art performance in many areas, they have the problem of vanishing gradient, i.e., when the network goes deeper, its performance gets saturated or even starts degrading rapidly. [17]. Because of this, the shallow counterparts can perform better than deep networks [17]. He *et al.* proposed residual blocks between layers to alleviate the problem of performance degradation [17]. These networks are called ResNets [18]–[22].

While the ideas of adding residual connections do exist, there has been very limited work that has applied it to unsupervised feature learning. Further, the existing work does not address the effect of performance degradation of deep neural networks for unsupervised feature learning. Therefore in this article, we present a framework that consists of residual blocks in AE architectures for unsupervised feature learning.

We use AEs to perform unsupervised feature learning. The unsupervised here refers to the unsupervised process of feature learning, i.e., learning of embedded representation from input data without using any labels. We used data labels only for the evaluation of learned embedded representations. We hypothesize that AEs with residual connections (RAE) will have improved resistance to performance degradation of learned features and improved feature learning capability compared to standard AEs. I.e., residual connections will alleviate possible information loss when increasing the number of hidden layers, and embedded representation will provide better separability for classification/clustering tasks.

Mainly for unlabeled data, it is challenging to decide the optimal number of hidden layers ahead when designing dimensionality reduction experiments. The proposed approach will always perform similar or better, even with a higher number of layers. Therefore, users have the advantage of designing few experiments with large networks, knowing that there is no adverse effect on the network's dimension reduction performance. To test our hypothesis, it is necessary to show that RAEs have lower performance degradation of unsupervised feature learning than AEs when increasing the networks' depth. We showed the effectiveness of the approach quantitatively by calculating the classification accuracy drop. I.e., we increased the number of hidden layers on both AEs and RAEs and checked how the classification accuracies on embedded representations change with the increase of the number of hidden layers. We used K Nearest Neighbor (KNN) for classification, as it allows us to check whether the same class samples are close to each other in the learned embedded representation (if the learned feature space learns a representation that encodes high-level concepts such as the classes of the input datasets).

The article presents the following contributions:

- Address the effect of performance degradation of deep neural networks for unsupervised feature learning
- Performance comparison between proposed architecture (RAE) and standard AE based feature learning, using a different number of hidden layers on three different datasets.
- Performance comparison between widely used unsupervised dimensionality reduction methods

We compare the presented method against two relevant groups of methods (a total of 7 different methods). The first group is represented by Autoencoders, which the literature indicates to be the most commonly used state-of-the-art deep learning based unsupervised dimensionality reduction architectures. We focus on standard Autoencoder and standard Convolutional Autoencoders because these are: 1) most frequently used; 2) other variants of AEs in the literature follow the principles of these two. Our objective was to evaluate how residual connections improve ''feature learning'', as such we compared against the same models with and without residual connections to evaluate improvement. The second group represents other types of feature extraction methods (five of those): Principal Component Analysis, Independent Component Analysis, Locally Linear Embedding, Factor Analysis, and Singular Value Decomposition.

The rest of the article is organized as follows: Section II presents the background and related work; Section III presents the RAE architecture; Section IV discusses the experiments and results; and finally, Section V presents the conclusions of the article.

II. BACKGROUND AND RELATED WORK

This section consists of three subsections. The first subsection discusses widely used traditional unsupervised dimensionality reduction techniques. The second section discusses Autoencoder based deep learning approaches for dimensionality reduction. The third section discusses the theory behind residual connections.

A. TRADITIONAL UNSUPERVISED MACHINE LEARNING FOR DIMENSIONALITY REDUCTION

As discussed in the introduction, feature learning is essential for efficient and accurate machine learning tasks. Two types of dimensionality reduction based feature learning techniques exist, namely feature selection and feature transformation [23]. A subset of features from the original space is selected in feature selection, whereas in feature transformation (Dimension reduction), it generates an entirely new set of features. Both try to keep as much information in the data as possible while reducing the dimension. However, feature selection can be misleading as it assigns weights to individual features ignoring the correlation between features [23]. Therefore, feature transformation approaches are preferable. Widely used such dimension reduction techniques are discussed below.

- *Principal Component Analysis (PCA):* A linear algorithm which preserves most of the data's variability in the latent space [15]. It minimizes the redundancy (measured through covariance) of data while maximizing information (Measured through variance) in the resulted space. Limitations include; 1) it only considers linear correlation, 2) input variables are assumed to be scaled at the numeric level [24].
- *Independent Component Analysis (ICA):* A linear transformation method that minimizes the dependence of the components of the transformed feature space [24]. Linearity is a major disadvantage of this method.
- *Locally Linear Embedding (LLE):* This is a non-linear algorithm that uses neighborhood preservation learning to generate subspace [15], [24]. However, this method has a high sensitivity for noise/outliers.
- *Factor Analysis:* This is the same as PCA in cases where the added noise is zero [25]. This method assumes that input data represent independent, random samples from a multivariate distribution. If variables are correlated, generated factors can be highly correlated [26].
- *Singular Value Decomposition (SVD):* This is mainly used for sparse data, i.e. when data contains many zero values. It converts the input data space to a latent representation with a reduced number of features while keeping the maximum information from the original space [27]. This approach is computationally expensive.

B. UNSUPERVISED DEEP AUTOENCODERS FOR DIMENSIONALITY REDUCTION

The traditional concept of unsupervised learning was mainly limited to the idea of data clustering and association rule mining. However, the expansion of deep learning methods and data mining combined with this era of big data has given a much broader perspective to traditional unsupervised learning. Therefore, unsupervised learning is used not only for clustering, but also for dimentionality reduction (also referred as unsupervised feature learning / deep embedded representation learning) [28], [29], generative modelling [30], [31], and auto-regressive modelling [32], [33]. This article focuses on deep unsupervised feature learning, which is the process of transforming the input space to an embedded space, preferably a lower dimension compared to the input data space, using deep neural networks.

Many recent classification tasks use different variants of AEs, to learn feature representation from high-dimensional input data, where the learned (extracted) features will provide good separability for classification tasks. In these cases, feature extraction will be performed in an unsupervised manner, whereas classification will be performed on the extracted features in the reduced dimension in a supervised manner. Feature learning using variants of AEs has shown the following advantages: improve the robustness of feature learning [13], non-linear feature extraction [12], replacing handcrafted features with efficient algorithms for

TABLE 1. Algorithm for Training the Proposed RAE.

Algorithm I: RAE Training
Inputs: Training set of images (X)
Outputs: Trained RAE, $Encoder$

Random Weight initialization
for each epoch e **do**
  for $i = 1...T$ **do** //number of training samples
    $x_i \leftarrow$ pick random input record from X
    $h \leftarrow x_i$
    for $l = 1...L_e$ **do** //each hidden layer l in encoder
      $h_f \leftarrow h$
      for j=1...F **do** //each layer j in f
        $h_f \leftarrow \sigma(W^{(l,j)}h_f + b^{(l,j)})$
      end for
      add residual connection to the hidden activation h_f
        $h \leftarrow W_r^{(l)}h + h_f$
    end for
    $y_i \leftarrow h$
    for $l = 1...L_d$ **do** //each hidden layer l in decoder do
      $y_i \leftarrow \sigma\left(V^{(l)}y_i + c^{(l)}\right)$
    end for
  end for
  Compute the reconstruction loss:
    $J_\theta = \frac{1}{T}\sum_{i=1}^{T}(x_i - y_i)^2$
  Perform one-step of the optimizer:
    $\theta = \text{argmin}_\theta(J_\theta)$
end for

Algorithm II: Deep Embedded Classification using KNN
Inputs: Training set(X), Training labels (Y), Testing set(X'), Testing labels (Y'), Trained $Encoder$
Outputs: Accuracy

$z_{train} \leftarrow Encoder(X)$ %convert training data to embedded representation
$z_{test} =\leftarrow Encoder(X')$ %convert testing data to embedded representation
Initialize a list (z_y) to store predicted class label
for each sample (i) in z_{test} do
  Initialize a list ($list$) to store $< distance, class >$ pairs
  $dist \leftarrow 0$, $label \leftarrow 0$
  for each sample (j) in z_{train} do
    $dist \leftarrow \|z_{test,i} - z_{train,j}\|$
    $label \leftarrow classlabelofhx$
    $list \leftarrow append < dist, label >$
  $class\_list \leftarrow$ find list of labels of K nearest neighbors
  $predicted\_class \leftarrow mode(class\_list)$
  $z_y =\leftarrow append(predicted\_class)$
  end for
end for
calculate accuracy using z_y and Y'

unsupervised feature learning [34], and reduces the time and storage space through dimensionality reduction [35].

The variant of deep AEs has been successfully used for deep embedded clustering tasks that perform feature learning and clustering simultaneously. In the past, clustering and feature learning were performed sequentially, i.e., it embeds the input space to a latent space and then performs clustering on the embedded space [29], [36]. With deep embedded clustering, it performs a joined optimization of feature learning (dimensionality reduction), and clustering [29]. For example, in [37], the authors have presented a deep clustering approach using fully connected convolutional AEs. They argue that the embedded representations extracted from an encoder may not be discriminative enough for efficient clustering. To overcome that, they have proposed a soft k -means model on top of the encoder to make a unified clustering model.

C. RESIDUAL CONNECTION WITHIN DEEP NEURAL NETWORKS

He *et al.* raised the awareness towards the problem of performance degradation [18]. I.e., when the network's depth increases, the network's performance will start to saturate, and eventually, it can even deteriorate [19]. This is not caused due to the over-fitting, but by the vanishing gradient of deep neural networks [19].

This problem has been addressed by various network designs networks such as ResNets [18], [20], Highway Networks [21], and DenseNets [22]. All these networks use the same design principle, i.e., skip connections or residual connections [19]. These networks with skip connections have consistently shown state-of-the-art performances in different neural network typologies [18], [21]. Other advantages of skip connection includes better easier training [19], numerical stability and easier optimization [19], [38]. Empirical evidence has shown that these deep architectures with skip connections should not produce a large error than their shallow counterparts [18], [20].

III. METHODOLOGY: ResNet AUTOENCODER BASED FEATURE LEARNING FOR DEEP EMBEDDED CLASSIFICATION

This section discusses the stacked ResNet Autoencoder (RAE) based feature learning approach for classification. Figure 2 presents the standard C-AE architecture with multiple convolution and max-pooling layers with multiple filters.

In this article, we implemented standard and convolutional AEs (AEs and C-AEs) with residual connections. Our intent was to convey the advantages of adding residual connection into AE networks to improve feature learning capability. Therefore, we designed a simple and reproducible experiment, which can run in a reasonable amount of time. We introduced residual connection into the AE architecture and presented the novel Residual Autoencoder (RAE) framework for deep embedded classification. We call its convolutional counterpart C-RAE. The proposed framework is presented in Figure 3 where (a) presents the training of presented RAE and (b) represent the classification task on learned features.

As similar to AEs, RAEs are trained to regenerate their inputs from its output (Figure 3 (a)). The input sample x is typically a n dimensional vector. Therefore the input layer consists of n neurons. Since the RAE network is trained to reconstruct the input, the output layer has the same number of neurons as the input layer. The hidden layers consist of m neurons.

Similar to AE, RAE also consists of two phases, i.e., encoding phase and decoding phase [39], [40]. For a high-dimentional input x, the encoder E computes a hidden representation $z = E(x)$. The decoder D reconstructs the hidden representation back to the high-dimensional input space $y = D(z)$. Both encoder and decoder have several hidden layers, making a deep (stacked) RAE.

For the decoder, each hidden layer is a non-linear mapping of the form $\sigma(Vz + c)$, where σ is an activation function such as sigmoid, tanh, softsign, or Relu [40]. V is the weight matrix. We use superscripts $V^{(l)}$ to denote the weight matrix that corresponds to layer l. In convolutional neural networks (C-RAE), the matrix multiplication is replaced by a convolution operation and max-pooling (see Fig. 3).

For the encoder, each hidden layer l is composed by a non-linear mapping $f(\cdot)$ and a residual connection $r(\cdot)$. Each

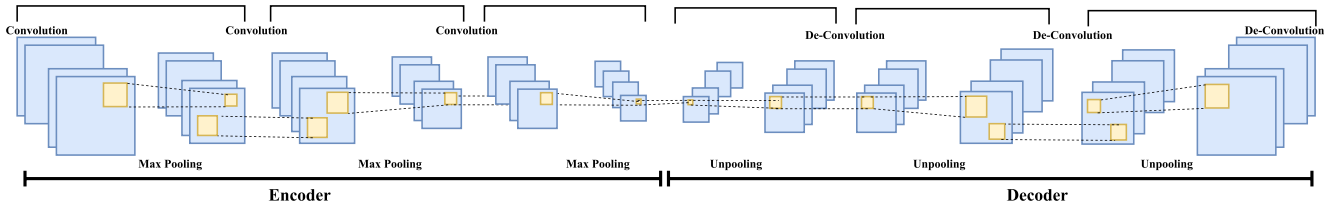


FIGURE 2. Standard architecture of stacked convolutional auto-encoder.

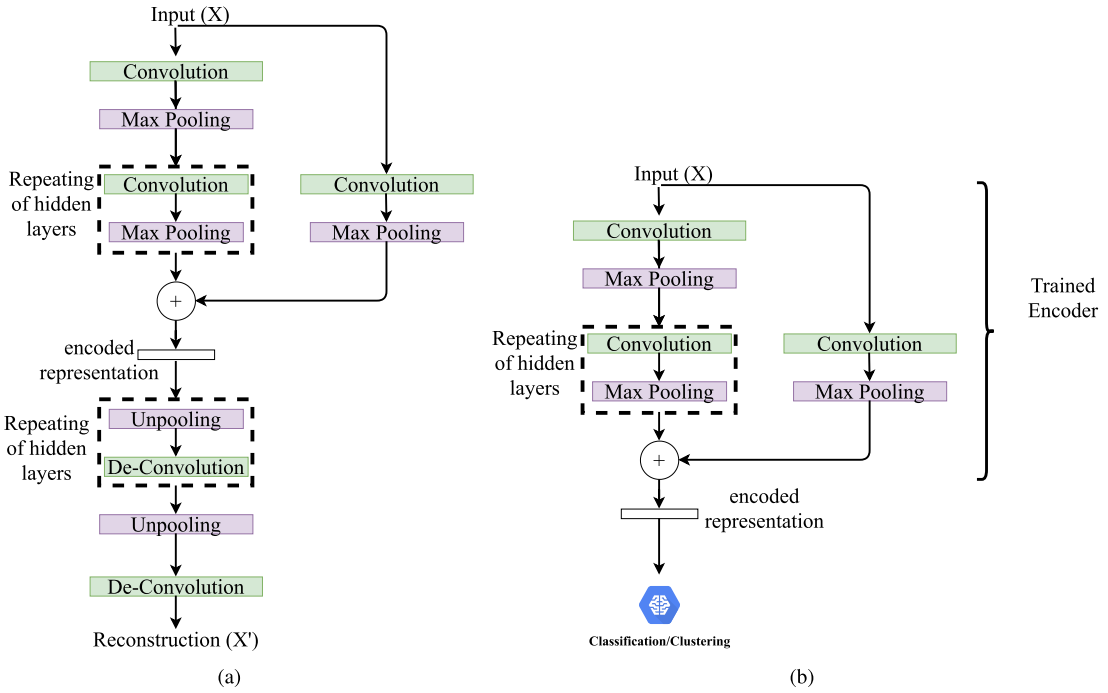


FIGURE 3. RAE based feature learning (a) Training of C-RAE, (b) C-RAE based classification/clustering.

hidden representation $h^{(l)}$ in a hidden layer l is computed follows:

$$h^{(l+1)} = r\left(h^{(l)}\right) + f\left(h^{(l)}\right) \tag{1}$$

The residual connection $r(h) = W_r h$ is a linear mapping that ensures the dimensions match the output of the function f. The function f can be thought of as a smaller network with F number of layers. Each layer in f is a non-linear mapping $\sigma(Wh + b)$, similar to the decoder layers. W is the weight matrix, and we use superscripts $W^{(l,j)}$ to denote the weight matrix that corresponds to layer l and sub-layer j. For C-RAEs, both matrix multiplications (W_r and W) are replaced by convolution operations and max pooling (see Fig. 3).

Similar to AE, the loss function J_θ of the RAE network is also computed using the difference between input(x) and the output(y), I.e. the error.

$$J_\theta = \frac{1}{T}\sum_{i=1}^{T} \|x_i - y_i\|^2 \tag{2}$$

where x_i is the ith input sample, y_i is the output for ith input sample, θ denotes the set of parameters of the autoencoder (weights and biases).

The RAE is trained to minimize the above loss function with T training samples using error-back-propagation. The pseudo-code for RAE training is presented in Algorithm I.

Similar to AE, the dimension of the hidden representations (z) of RAE can be smaller or larger than the dimension of x. When the hidden representation is small, the RAE performs dimensionality reduction (data compression) [40].

The encoded value z is viewed as the extracted feature or the hidden representation for the input data. Once the encoder converts the input samples (x) to an embedded representation z, then classification or clustering can be performed on this latent space (shown in Figure 3 (b)).

For classification purposes, any supervised classification algorithm can be integrated at the end of the encoder (Figure 3 (b)). For this experiment, the K-Nearest Neighbor algorithm (KNN) is used. Algorithm II presents the KNN based deep embedded classification.

As presented in Algorithm II, the trained RAE's encoder is used to generate an embedded representation of train and test data (line 1-2). Then class labels for test data can be predicted by comparing each test record with all the train records and find the mode class label of K nearest train records (Algorithm II line 4-12). The distance between a test record and a train record should be calculated using a distance calculation method to find the nearest neighbors. For this experiment, Euclidean distance is calculated:

$$dist(z_{test}, z_{train}) = \sqrt{\sum_{i=0}^{dim} \left(z_{test,i} - z_{train,i}\right)} \quad (3)$$

where z_{test} is the test record, z_{train} is the train record, and dim is the dimension of the embedded feature space (z). However, it is possible to use other distance calculation methods such as Minkowski, Manhattan, Mahalanobis, and cosine. Finally, predicted labels and actual labels are compared to calculate the accuracy of the KNN algorithm.

IV. EXPERIMENT AND RESULTS

This section discusses the experiments and results. First, we discuss the datasets used for experimental evaluation. Then, we present the experimental set-up and architecture details of the networks. Finally, we discuss the results of the experiment with a comparison between existing dimensionality reduction methods.

A. DATASETS

Three datasets were used for experimental evaluation: 1) MNIST [41], 2) CIFAR10 [42], and 3) Fashion MNIST [43]. All the datasets were scaled to the 0-1 range. These benchmark datasets were selected due to their relatively high dimension and reasonable training time with deep networks. Datasets were directly obtained from the Keras library [44].

The **MNIST dataset** consists of hand-written digits (0-9), where each digit is an image of 28 X 28 pixels in size. The complete MNIST dataset was used, which consist of 55000 train images and 10000 test images.

The **Fashion MNIST dataset** benchmark dataset consist of images used for clothing classification. It consists of images with 28 X 28 pixels in size. Class labels include (T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot) The complete Fashion MNIST dataset was used, which consists of 60000 train images and 10000 test images. Images belong to 10 classes.

The **CIFAR10 dataset** consists of color images of 32 X 32 pixels in size. These images correspond to 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). The complete CIFAR10 dataset was used, which consist of 50000 train images and 10000 test images.

B. HYPER-PARAMETERS AND ARCHITECTURAL DETAILS

To maintain consistency in the experiments, all the architectures were kept constant across datasets when increasing the number of layers. Only two filters were used with size 32 and 64. The size of the embedded representation is kept at 32. The number of layers were increased by repeating the convolution layer and pooling layer for a given filter size. For this experiment number of repeating layers were increased from 2 to 90 for each filter. Optimizer (adadelta) and K(5) were kept constant for all the experiments across datasets. Batch normalization and leakyRelu was used to improve model performance. For illustration purposes, the MNIST dataset architecture with two filters (32,64) and 2 repeats is presented in Figure 4. For a given number of repeats (f), the total number of hidden layers is 2 + (f*no. of filters).

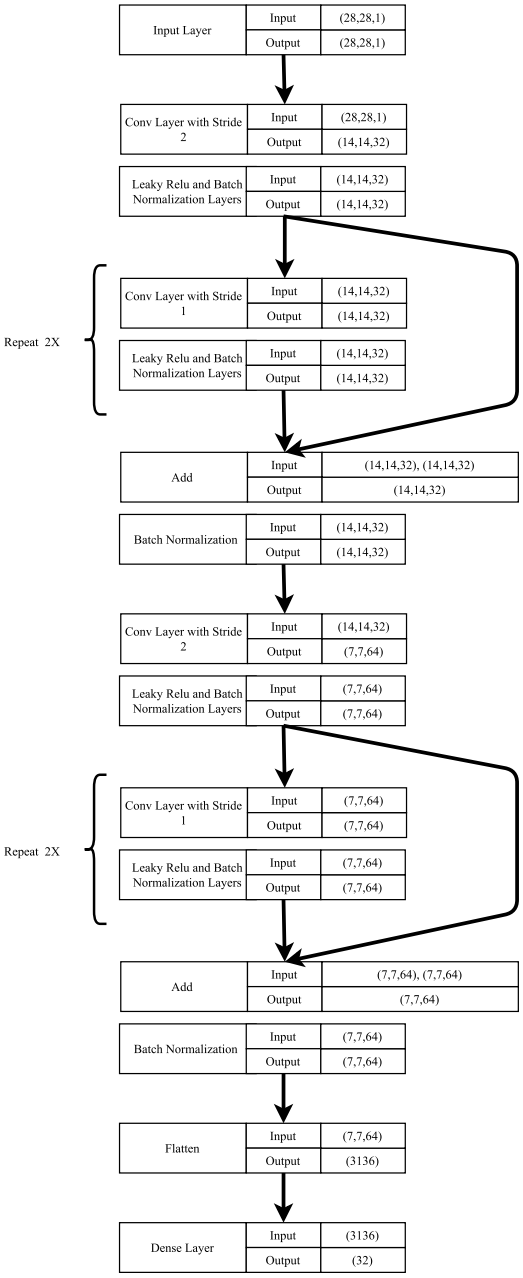


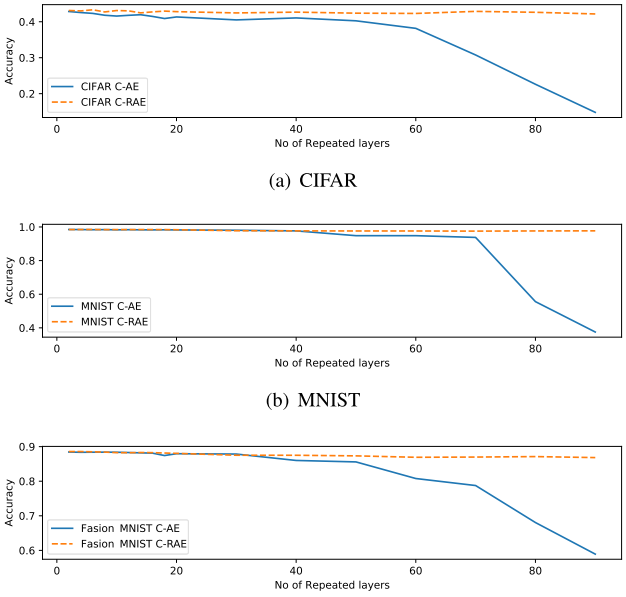
FIGURE 4. Architecture.

TABLE 2. Classification Accuracies of Models for Different Datasets.

Dataset	Model	No of Repeated Layers											Performance Degradation
		2	6	10	20	30	40	50	60	70	80	90	
MNIST	*C-AE*	0.9849	0.9836	0.9830	0.9824	0.9805	0.9767	0.9806	0.9482	0.9379	0.5559	0.3752	61.90
	C-RAE	0.9846	**0.9853**	0.9845	0.9835	0.9776	0.9768	0.9762	0.9761	0.9753	0.9764	0.9770	0.86
CIFAR	*C-AE*	0.4285	0.4233	0.4158	0.4134	0.4052	0.4107	0.4026	0.3817	0.3076	0.2262	0.1480	65.46
	C-RAE	0.4313	**0.4333**	0.4312	0.4281	0.4246	0.4268	0.4239	0.4231	0.4289	0.4263	0.4217	2.68
Fashion MNIST	*C-AE*	0.8844	0.8839	0.8838	0.8795	0.8785	0.8600	0.8558	0.8078	0.7875	0.6805	0.5892	33.38
	C-RAE	**0.8858**	0.8850	0.8826	0.8805	0.8751	0.8750	0.8733	0.8692	0.8698	0.8712	0.8683	1.97

TABLE 3. Comparative Analysis.

Dataset	KNN applied after unsupervised feature extraction (Embedded classification)									KNN on original high-dimensional feature space
	AE	RAE	C-AE	C-RAE	PCA	Standard LLE	ICA Embedding	Factor Analysis Embedding	Truncated SVD Embedding	
MNIST	0.9745	0.9758	0.9849	**0.9853**	0.9758	0.9684	0.9713	0.9621	0.9755	0.9688
Fashion MNIST	0.8599	0.8617	0.8844	**0.8858**	0.8524	0.8126	0.8538	0.8499	0.8517	0.8552
CIFAR10	0.4182	0.4201	0.4285	**0.4333**	0.4039	0.2831	0.4134	0.4070	0.4019	0.3398



(a) CIFAR

(b) MNIST

(c) Fashion MNIST

FIGURE 5. Classification accuracy vs number of hidden layers.

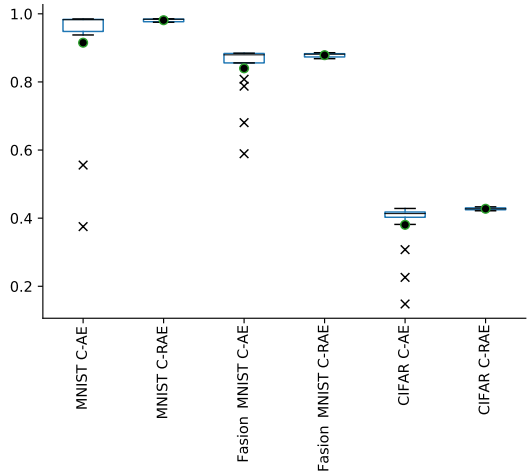


FIGURE 6. Accuracy distribution.

C. CLASSIFICATION ACCURACY

The trained autoencoder models were used to generate the embedded representation for the datasets. These embedded representations were used for the classification using the KNN algorithm, i.e., encoder followed by KNN used as the classification network. Each experiment was repeated five times, and the average performances were recorded.

Table 2 shows the deep embedded classification accuracy obtained using the two models, C-AE and C-RAE, for different datasets when increasing the number of hidden layers. When comparing all the models, C-RAE showed improved accuracy compared to C-AE for all three datasets (highlighted values in Table 2). Table 3, column 6 shows the classification performance of KNN on the original high dimensional data. It can be seen that both C-AE and C-RAE based deep embedded classification showed better accuracies than just applying KNN on original data. This infers that these deep neural network models convert original data into embedded representations that are more suitable than using the original input data for down-stream tasks such as classification.

Figure 5 shows a plot of the accuracies against no of repeated layers. When increasing the number of layers, a small fluctuation of accuracy was observed for small models (up to 20 repeated layers) for all the datasets. For large models, when increasing the no of hidden layers, the accuracies started to decrease. However, C-RAE showed significantly lower degradation compared to C-AE. Therefore, it can be inferred that C-RAE based embedded representations are less likely to under-perform when increasing the number of layers.

Figure 6 shows classification accuracy distribution in box and whisker graphs for all three datasets when increasing the number of layers. The height of the box plot indicates the variability of classification accuracy for each model. Anything outside the normal distribution is marked as outliers shown as "X" marks. A shorter box and whiskers plot indicates low variability of classification accuracies. For all the

C-RAE, the whiskers are shorter than C-AEs, and there are no outliers. It shows that C-RAE has consistent performance with low variability when increasing the number of layers. Mean values are marked with ''O''. All C-RAEs mean values are higher than the C-AEs. These observations show that with the change of the number of hidden layers, C-RAEs have consistent performance, whereas, for standard C-AEs, a thorough cross-validation process is needed.

The last column of Table 2 shows the overall performance degradation for deep embedded classification when increasing the number of hidden layers. The performance degradation (PD) was calculated as the percentage accuracy drop when increasing the number of layers:

$$PD = \frac{(MaximumAcc - MinimumAcc) * 100}{MaximumAcc} \tag{4}$$

Both C-RAE and C-AE showed some performance degradation for all three datasets. C-AE without residual connection showed 33.38% - 65.46% performance degradation whereas C-RAE showed 0.86% - 1.97% performance degradation. Based on the experimental result, it can be seen that residual connections reduce possible performance degradation significantly.

D. COMPARISON BETWEEN WIDELY USED DIMENSIONALITY REDUCTION METHODS

Table 3 presents the performance comparison between proposed approaches and widely used unsupervised dimensionality reduction methods. We compared the proposed approach with two state-of-the-art deep neural network based dimensionality reduction methods (AE and C-AE) and five most widely used conventional dimensionality reduction methods in the recent literature (PCA, LLE, ICA, Factor Analysis embedding, Truncated SVD embedding). As described in the previous section, all these methods were used to convert the high dimensional input space to an embedded representation of 32 features. Then, KNN was used to perform the classification on the embedded representations. Further, KNN was ran to calculate the classification accuracy on the original high dimensional space (last column of Table 3). For MNIST, all the embedded classification approaches except LLE and FAE showed better accuracies compared to applying KNN on the original high dimensional feature space. C-RAE showed the highest accuracy (.9853) for MNIST. For Fashion MNIST, only deep neural network based embedded classification showed higher accuracy compared to KNN. C-RAE showed the highest accuracy (0.8858) for Fashion MNIST. For CIFAR10, all the embedded classification methods except LLE showed higher accuracy compared to KNN. C-RAE showed the highest accuracy (0.4333) for CIFAR10. When comparing AEs and RAEs on all three datasets, RAEs showed slightly better performance. When comparing RAE and C-RAE, C-RAE showed better accuracy on all three datasets. The results of 2 and Table 3 infers that deep neural network models convert original data into embedded representations that are more suitable than using the original input data for down-stream tasks such as classification, and C-RAE based embedded representations are less likely to under-perform when increasing the number of layers.

E. OVERALL DISCUSSION AND FUTURE WORK

Our hypothesis was that when adding new layers to standard AEs, their ability for effective feature learning degrades. Through accuracy comparison in Table 2, we confirmed that addition of residual connections to AEs (RAEs), improved their overall classification accuracy without incurring significant performance degradation (relative to standard AEs).

Through a comprehensive comparison of widely used unsupervised dimensionality reduction methods in Table 3, we demonstrated that the C-RAE outperforms widely used feature learning methods such as standard AE, KNN, PCA, LLE, ICA, Factor Analysis, and SVD by 1%-3% improvements of classification accuracy. In addition to the accuracy improvement over standard CAE, C-RAE showed significantly lower performance degradation of classification accuracy (less than 3%) compared to CAE (33%-65%), when increasing the network depth. These results evidenced the advantages and the overall superiority of C-RAEs for unsupervised feature learning compared to standard AEs and widely used traditional methods.

Finally, by implementing the novel RAE framework presenting here, one does not need to go through a trial and error process of finding the best architecture. Instead, one can safely go with more layers in case a more complex model is required for improved overall performance while not sacrificing the dimensionality reduction performance.

The experiment was tested using three datasets that can be trained with deep neural networks within a reasonable amount of time. However, it has to be noticed that the advantage of using a deep neural network is more prominent when dealing with more complex datasets. Therefore, in future work, the framework will be tested with more complex datasets, which are high in dimension and number of data records.

V. CONCLUSION

In this article, we tackle the performance degradation problem of automated deep unsupervised feature learning. We introduced an unsupervised deep learning framework, consisting of ResNet Autoencoder (RAE) and its convolutional version C-RAE, that allows making deeper neural networks while not sacrificing its dimensionality reduction performance. In this way, we improve resistance to performance degradation compared to standard Autoencoders (AEs) for feature learning. The performance of RAE on learning deep embedded representations was evaluated on a classification task using KNN. RAE was compared against AE while increasing the number of hidden layers. We did this comparison on three benchmark datasets. We demonstrated that C-RAE showed the highest accuracy on all three datasets. At the same time, C-RAE based classification only showed 0.86% to 2.68% performance degradation, which is signif-

icantly lower than the performance degradation showed by standard C-AE (33.38% - 65.46%). The empirical results confirmed that RAE reduces performance degradation of deep embedded representation based classification. This framework allows users to design fever number of experiments knowing that larger networks will not affect the network performance, especially when dealing with unlabelled data where the optimal network size is challenging to decide. Further, the classification accuracy distribution showed that RAE models perform better in terms of mean accuracy and accuracy variance (low variance), making them more suitable for deep embedded classification tasks than AE. Finally, we compared RAEs with widely used dimensionality reduction methods and showed that C-RAE outperforms on all experimented datasets.

REFERENCES

[1] Q. Zhang, L. T. Yang, Z. Yan, Z. Chen, and P. Li, "An efficient deep learning model to predict cloud workload for industry informatics," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3170–3178, Jul. 2018.

[2] K. Kailing, H.-P. Kriegel, and P. Kröger, "Density-connected subspace clustering for high-dimensional data," in *Proc. SIAM Int. Conf. Data Mining*, FL, USA, 2014, pp. 246–256.

[3] C. S. Wickramasinghe, D. L. Marino, K. Amarasinghe, and M. Manic, "Generalization of deep learning for cyber-physical system security: A survey," in *Proc. 44th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2018, pp. 745–751.

[4] Y. Xu, Y. Sun, J. Wan, X. Liu, and Z. Song, "Industrial big data for fault diagnosis: Taxonomy, review, and applications," *IEEE Access*, vol. 5, pp. 17368–17380, 2017.

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

[6] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A tensor-train deep computation model for industry informatics big data feature learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3197–3204, Jul. 2018.

[7] M. Kang, M. R. Islam, J. Kim, J.-M. Kim, and M. Pecht, "A hybrid feature selection scheme for reducing diagnostic performance deterioration caused by outliers in data-driven diagnostics," *IEEE Trans. Ind. Electron.*, vol. 63, no. 5, pp. 3299–3310, May 2016.

[8] Y. Ren, K. Hu, X. Dai, L. Pan, S. C. H. Hoi, and Z. Xu, "Semi-supervised deep embedded clustering," *Neurocomputing*, vol. 325, pp. 121–130, Jan. 2019.

[9] J. Zabalza, J. Ren, J. Zheng, H. Zhao, C. Qing, Z. Yang, P. Du, and S. Marshall, "Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging," *Neurocomputing*, vol. 185, pp. 1–10, Apr. 2016.

[10] A. Hinrichs, J. Prochno, and M. Ullrich, "The curse of dimensionality for numerical integration on general domains," *J. Complex.*, vol. 50, pp. 25–42, Feb. 2019.

[11] D. Zhang, Y. Sun, B. Eriksson, and L. Balzano, "Deep unsupervised clustering using mixture of autoencoders," *CoRR*, vol. abs/1712.07788, pp. 1–8, Dec. 2017.

[12] C. Xing, L. Ma, and X. Yang, "Stacked denoise autoencoder based feature extraction and classification for hyperspectral images," *J. Sensors*, vol. 2016, pp. 1–10, Jun. 2016.

[13] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, Jul. 2016.

[14] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognit. Lett.*, vol. 42, pp. 11–24, Jun. 2014.

[15] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, Apr. 2016.

[16] L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, *Scaling Learning Algorithms Toward AI*. Mainz, Germany: MITP, 2007, pp. 321–359.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, pp. 1–9, Jun. 2015.

[19] A. Zaeemzadeh, N. Rahnavard, and M. Shah, "Norm-preservation: Why residual networks can become extremely deep?" *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Apr. 27, 2020, doi: 10.1109/TPAMI.2020.2990339.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *CoRR*, vol. abs/1603.05027, pp. 630–645, Sep. 2016.

[21] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 2. Cambridge, MA, USA: MIT Press, 2015, pp. 2377–2385.

[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[23] M. Pechenizkiy, "The impact of feature extraction on the performance of a classifier: kNN, Naïve Bayes and C4.5," in *Advances in Artificial Intelligence*, B. Kégl and G. Lapalme, Eds. Berlin, Germany: Springer, 2005, pp. 268–279.

[24] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Proc. Sci. Inf. Conf.*, Aug. 2014, pp. 372–378.

[25] H. S. Park, R. Dailey, and D. Lemus, "The use of exploratory factor analysis and principal components analysis in communication research," *Hum. Commun. Res.*, vol. 28, no. 4, pp. 562–577, Oct. 2002.

[26] C. Reimann, P. Filzmoser, and R. G. Garrett, "Factor analysis applied to regional geochemical data: Problems and possibilities," *Appl. Geochemistry*, vol. 17, no. 3, pp. 185–206, Mar. 2002.

[27] P. C. Hansen, "The truncatedSVD as a method for regularization," *BIT, Numer. Math.*, vol. 27, no. 4, pp. 534–553, Dec. 1987.

[28] A. A. Mohamed, "An effective dimension reduction algorithm for clustering arabic text," *Egyptian Informat. J.*, vol. 21, no. 1, pp. 1–5, Mar. 2020.

[29] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers, "Clustering with deep learning: Taxonomy and new methods," Jan. 2018, *arXiv:1801.07648*. [Online]. Available: http://arxiv.org/abs/1801.07648

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2014, pp. 2672–2680.

[31] I. J. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," *CoRR*, vol. abs/1701.00160, pp. 1–57, Dec. 2017.

[32] P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. Lecun, "Predicting deeper into the future of semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 648–657.

[33] G.-Y. Chen, M. Gan, and G.-L. Chen, "Generalized exponential autoregressive models for nonlinear time series: Stationarity, estimation and applications," *Inf. Sci.*, vol. 438, pp. 46–57, Apr. 2018.

[34] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Process.*, vol. 130, pp. 377–388, Jan. 2017.

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

[36] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*. New York, NY, USA: ACM, 2004, p. 29.

[37] F. Li, H. Qiao, and B. Zhang, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *Pattern Recognit.*, vol. 83, pp. 161–173, Nov. 2018.

[38] D. Balduzzi, M. Frean, L. Leary, J. Lewis, W.-D. Ma, and B. Mcwilliams, "The shattered gradients problem: If resnets are the answer, then what is the question?" in *Proc. Int. Conf. Mach. Learn.*, Feb. 2017, pp. 342–350.

[39] A. I. Károly, R. Fullér, and P. Galambos, "Unsupervised clustering for deep learning: A tutorial survey," *Acta Polytechnica Hungarica*, vol. 15, pp. 29–53, Dec. 2018.

[40] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.

[41] Y. Lecun. *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/ and https://ci.nii.ac.jp/naid/10027939599/en/

[42] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., May 2012.

[43] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, pp. 1–6, Dec. 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[44] F. Chollet, *Keras*. San Francisco, CA, USA: GitHub, 2015.

CHATHURIKA S. WICKRAMASINGHE received the B.Sc. degree in computer science from the University of Peradeniya, Sri Lanka, in 2016. She is currently pursuing the Ph.D. degree in computer science with Virginia Commonwealth University, Richmond. She is currently a Research Assistant. Her research interests include machine learning, unsupervised learning, explainable AI, generalization, and visual data mining.

DANIEL L. MARINO received the B.Eng. degree in automation engineering from La Salle University, Colombia, in 2015. He is currently pursuing the Ph.D. degree with Virginia Commonwealth University. He is currently a Research Assistant with Virginia Commonwealth University. His research interests include stochastic modeling, deep learning, and optimal control.

MILOS MANIC (Fellow, IEEE) is currently a Professor with the Computer Science Department and also the Director of VCU Cybersecurity Center, Virginia Commonwealth University. He completed more than 40 research grants in data mining and machine learning applied to cyber security, critical infrastructure protection, energy security, and resilient intelligent control. He has given more than 40 invited talks around the world, authored more than 200 refereed articles in international journals, books, and conferences, holds several U.S. patents. He received the 2018 R&D 100 Award for Autonomic Intelligent Cyber Sensor (AICS), one of top 100 science and technology worldwide innovations in 2018. He is an inductee of U.S. National Academy of Inventors (class of 2019) and a Fellow of Commonwealth Cyber Initiative (specialty in AI & Cybersecurity).

He was a recipient of the 2012 J. David Irwin Early Career Award, the 2017 IEM Best Paper Award, and the IEEE IES 2019 Anthony J.Hornfeck Service Award. He served as a Founding Chair for the IEEE IES Technical Committee on Resilience and Security in Industry and a General Chair for IEEE IECON 2018 and IEEE HSI 2019. He serves as an Associate Editor for the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE OPEN JOURNAL OF INDUSTRIAL ELECTRONICS SOCIETY, and is IES Officer and Senior AdCom member. He served as an Associate Editor for the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.