Selective Deep Autoencoder for Unsupervised Feature Selection

Wael Hassanieh, Abdallah Chehade

Department of Industrial & Manufacturing Systems Engineering
University of Michigan-Dearborn
4901 Evergreen Road, Dearborn, Michigan 48128 USA
waelh@umich.edu, achehade@umich.edu

Abstract

In light of the advances in big data, high-dimensional datasets are often encountered. Incorporating them into data-driven models can enhance performance; however, this comes at the cost of high computation and the risk of overfitting, particularly due to abundant redundant features. Identifying an informative subset of the features helps in reducing the dimensionality and enhancing model interpretability. In this paper, we propose a novel framework for unsupervised feature selection, called Selective Deep Auto-Encoder (SDAE). It aims to reduce the number of features used in unlabeled datasets without compromising the quality of information obtained. It achieves this by selecting sufficient features - from the original feature set - capable of representing the entire feature space and reconstructing them. Architecturally, it leverages the use of highly nonlinear latent representations in deep Autoencoders and intrinsically learns, in an unsupervised fashion, the relevant and globally representative subset of features through a customized Selective Layer. Extensive experimental results on three high-dimensional public datasets have shown promising feature selection performance by SDAE in comparison to other existing state-of-the-art unsupervised feature selection methods.

Introduction

In light of the advances in big data, high-dimensional datasets are often encountered. Deploying them in data-driven models can boost performance; however, this comes at the cost of high computation and poor out-of-training accuracy (possibly due to poor robustness and/or overfitting) (Goodfellow, Shlens, and Szegedy 2014). The presence of a large number of redundant features is the main problem. Moreover, such models are prone to low interpretability, in a world where humans and industries are increasingly demanding to understand the data and models they are working with (Doshi-Velez and Kim 2017). It is evident in the Computer Vision field in which high-dimensional images contain a significant number of redundant pixels (Bolón-Canedo and Remeseiro 2020). Elsewhere, connected devices governed by the Internet of Things (IoT) generate massive amounts of data that can be overwhelming for practitioners who try to extract meaningful insights from abundant sensor readings (Dièn et al. 2020). This is where feature selection thrives.

Feature selection (FS) is a learning process that selects features that are significant according to the task at hand and eliminates the others. It is often used to enhance data explainability, improve learning model performance, and cut off computational time (Yu and Liu 2003). It is important to highlight the difference between feature selection and dimensionality reduction. The latter, which includes methods like Principal Component Analysis (PCA) (Hotelling 1933) and Autoencoders (Hinton and Salakhutdinov 2006), is used to represent data in a new lower dimensional space, rather than select the important features from the original feature space. There exist various feature selection categories: supervised (Robnik-Šikonja and Kononenko 2003; Guyon and Elisseeff 2003) and unsupervised methods (Wang et al. 2013; Alelyani, Tang, and Liu 2018). The latter is often preferred because labels, which might be time-consuming and/or costly to obtain depending on the application, are not required. The literature presents several types of unsupervised feature selection: Filter (Li et al. 2018) (e.g. Laplacian Score (LS) (He, Cai, and Niyogi 2005) and Spectral Feature Selection (Zhao and Liu 2007)), Wrapper (e.g. AMBER (Ramjee and Gamal 2020)) and Embedded based methods. The embedded feature selection methods are where recent state-of-the-art (SOTA) methods are developed, e.g. Deep Feature Selection (Li, Chen, and Wasserman 2016) and UDFS algorithm (Yang et al. 2011) which applies L2-L1 regularization on weights of inputs to select from them useful features. Similarly, the MCFS algorithm (Cai, Zhang, and He 2010) also uses regularization to keep features that maintain clustering structures. Unsupervised feature selection methods have faced two main limitations: the first is the challenge of searching for possible important subsets of features when labels are not present, and the second is taking into account the inter-correlation of features. In particular, the set of selected features needs to be "globally representative and diverse" (Wu and Cheng 2021).

Deep Neural Network (DNN) learning architectures have witnessed a revival of interest in the 2000s due to the discovery of approaches capable of learning the parameters. AutoEncoders (AE), which have been part of the neural network space for decades, were developed for the purpose of unsupervised dimensionality reduction, mainly as a nonlinear extension of PCA (Kramer 1991). It consists of two sequential neural network parts: an Encoder and a Decoder.

The former maps a set of real features into a higher-level latent encoding space, and the latter recovers the original feature space from the encoding space (Pinaya et al. 2020). Leveraging this complex latent space has proved to be essential and advantageous in diverse applications (Savargaonkar et al. 2022). Although AEs were mainly used for dimensionality reduction or creating a better representative latent representation of the input (specifically in the bottleneck layer), many in the literature have tried to expand its use towards feature selection, e.g. classification of anticancer drug response (Xu et al. 2019). In the last few years, several feature selection methods based on AEs have emerged. Multiplicative Autoencoder based on Feature Selection (MAFS) (Chandra and Sharma 2015) is based on a shallow AE incorporating a custom multiplicative aggregation function with masking on the input layer to evaluate redundancy; the masking is influenced by work on Stacked Denoising AEs (Vincent et al. 2010). AutoEncoder Feature Selector(AEFS) (Han et al. 2018) applies ℓ_2-ℓ_1 regularization on the encoder's weights to get a subset of important features. Agnostic Feature Selection (Doquet and Sebag 2020) introduces AgnoS-S which is a shallow AE with a first layer of slack variables that have an ℓ_1-norm applied to them for the purpose of selecting features from the input layer. As seen in the mentioned three pieces of literature, regularization in AEs is essential for feature selection; that's because AEs are said to learn to reconstruct or copy the input to the output; however, to learn something useful other than reconstruction, regularization should be present (Goodfellow, Bengio, and Courville 2016). More recently, Concrete Autoencoders (CAE) (Abid, Balin, and Zou 2019) inserts to the AE a custom first hidden layer, the "concrete selector layer", which picks the features according to a high probability of connecting to the input/feature nodes. Even though this method obtains high performance, however, features selected through CAE are not validated or consistent over multiple runs. Lately, Fractal Autoencoders (FAE) (Wu and Cheng 2021) attempted to tackle the global representation and diversity challenges; it consists of a shallow AE, which contains a one-to-one layer on the input (much like the slack variables concept in AgnoS-S (Doquet and Sebag 2020)) while training it with a sub-DNN of similar architecture but with only k selected features. A competing interest arises during training between reconstruction and obtaining the best representative k features. Moreover, these mentioned methods all are built on shallow AEs, which should make it difficult to select features when the set of features is complex and not linearly correlated.

In this paper, we propose a novel Autoencoder-based unsupervised framework, called Selective Deep AutoEncoder (SDAE), that leverages the strength of deep AEs for feature subset selection and imputation. Unlike, other methods in the literature, it does not involve selecting a specific number of features before training; it intrinsically learns the sufficient subset of relevant features to represent the whole feature space.

The main contributions of the paper can be summarized into the following:

- In this work, we propose a novel unsupervised feature selection approach, SDAE. It trains a deep AE to learn representative latent representations. SDAE then activates a Selective Layer to select the minimal feature subset that accurately reconstructs the latent representations.
- Most SOTA methods for unsupervised feature selection are limited to a single bottleneck hidden layer. In contrast, SDAE allows deeper architectures that are proven to be necessary for complex applications. We show the significant performance advantage of the deeper architecture of SDAE over the other approaches on a highly non-linear simulated dataset, as well as a commonly used dataset for feature selection.
- Unlike most SOTA methods for unsupervised feature selection, SDAE does not involve selecting a specific number of features before training; it intrinsically learns the sufficient subset of relevant original features to represent the whole original feature space.
- We validate and benchmark SDAE with extensive experiments on three real datasets.

The notations and definitions are given as follows. Let N, n, k, and p be the numbers of samples, input dimension/features, selected features, and encodings dimensions, respectively. Let $\mathbf{X} \in \mathbb{R}^{N \times n}$ be a matrix containing the high-dimensional input data.

The remainder of the paper is organized as follows. We first introduce the methodology, where the training algorithm of SDAE is thoroughly explained. Then, we show its application in an experimental case study, characterized by non-linearly correlated features. Afterward, we benchmark the framework against other SOTA methods evaluated on public datasets.

Proposed Approach

In this section, we will present the formulation and architecture of the Selective Deep AutoEncoder (SDAE).

Approach Overview

The SDAE is a SOTA Feature Selection framework that leverages the AutoEncoder (AE) structure in selecting sufficient features capable of reconstructing the whole original feature space. The framework consists of the Training and Inference stages. In the former, the learned model simultaneously aims at obtaining good reconstructions of the whole feature space and selecting sufficient features necessary for maintaining the reconstructions. In the Inference stage, an SDAE network is assembled which is capable of constructing the original feature space from select features. The full framework is depicted in Figure 1. In the following, we will further explain the architecture in detail.

Deep Autoencoders for Reconstruction Formalization

The AE consists of: the Encoder $f(\cdot)$, and the Decoder $g(\cdot)$. It is optimized for reconstruction when trained to minimize the following:

$$\min_{\theta=\{\theta_{enc},\theta_{dec}\}} \|\mathbf{X} - g(f(\mathbf{X}, \theta_{enc}), \theta_{dec})\|_F^2 \qquad (1)$$

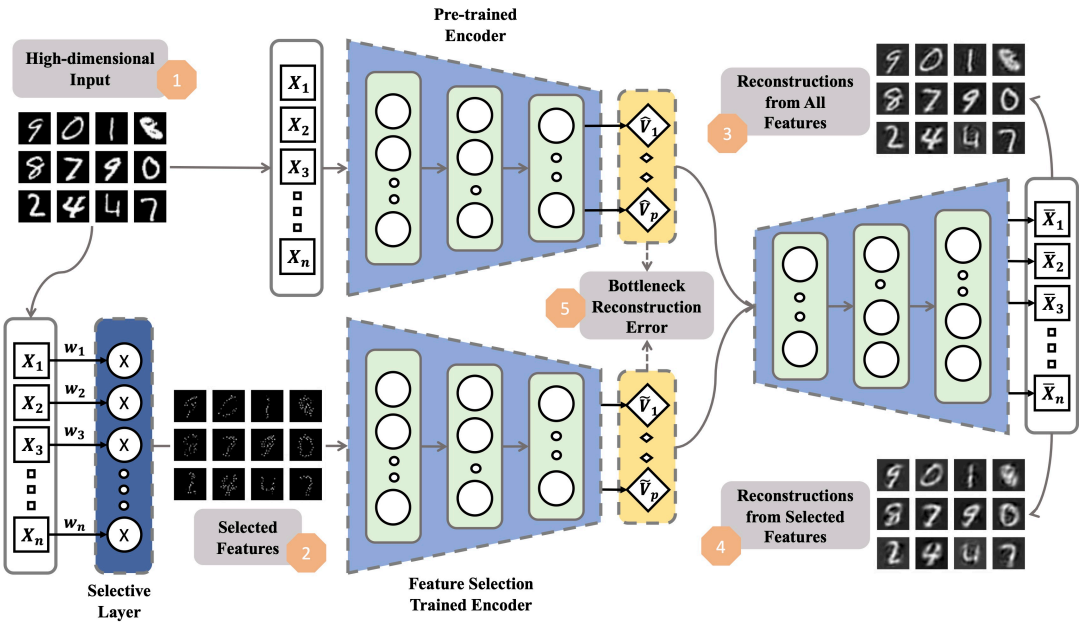

Figure 1: SDAE Architecture Training Framework. 1) The whole feature space, $\mathbf{X}$. 2) Selected features $\mathbf{X}_{\mathcal{F}_k}$. 3) Reconstruction from the whole feature space $g(f(\mathbf{X}, \theta_{enc}), \theta_{dec})$. 4) Reconstruction from selected features $g(f(\mathbf{X}_{\mathcal{F}_k}, \theta_{enc}), \theta_{dec})$. 5) Encodings reconstruction error $\|\hat{\mathbf{V}} - \tilde{\mathbf{V}}\|_{\mathrm{F}}^2$.

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. θ_{enc} is the set of weights connecting the layers in the Encoder, while θ_{dec} is the set of weights connecting the layers in the Decoder. The Deep AE consists of several dense fully-connected layers. $f(\mathbf{X}, \theta_{enc})$ encodes the input data into a latent representation $\mathbf{V} \in \mathbb{R}^{N \times p}$, with p being the encodings dimension, i.e. the bottleneck layer's dimension. The encodings present an equivalent representation of the feature space but in a different dimension p; if the AE is undercomplete, then the encodings represent a lower-dimensional representation of the original feature space (Goodfellow, Bengio, and Courville 2016). However, the contents of a transformed sample by $f(\cdot)$ are not meaningful in the latent space. The Decoder transforms the encodings via $g(\mathbf{V}, \theta_{dec})$ back to the original feature space to reconstruct the same inputted samples.

Problem Formulation: Unsupervised Feature Selection

In feature selection, the goal is to identify a subset of informative features from the original feature space that is considered globally representative of that space. It is considered unsupervised when the set of features is not collected for the purpose of predicting/estimating a label/target. In other words, to select a subset of features and consider them sufficiently informative of the whole feature space, the k selected features should be able to reconstruct the whole feature space as per:

$$\min_{\mathcal{F}_k, \theta} \|\mathbf{X} - F(\mathbf{X}_{\mathcal{F}_k}, \theta)\|_{\mathrm{F}}^2 \tag{2}$$

where $\mathcal{F}_k$ represents the subset of the selected k features, $\mathbf{X}_{\mathcal{F}_k} \in \mathbb{R}^{N \times k}$ is the subset of $\mathbf{X}$ based on selecting the columns corresponding to the features $\mathcal{F}_k$, F is a deep learning module that maps $\mathbb{R}^{N \times k}$ to $\mathbb{R}^{N \times n}$. It is shown in literature (Hamo and Markovitch 2005) that the optimization problem in (2) is an NP-hard problem. Accordingly, this paper proposes an algorithm that presents a sub-optimal solution for the unsupervised feature selection problem.

Selective Deep Autoencoder (SDAE): Training

In our unsupervised feature selection problem statement, we have two main purposes: first, identify a lower latent dimensional space that is globally representative of the original higher-dimensional feature space; second, learn an informative subset of features that is capable of learning the latent lower dimensional space. However, it should be noted that these are two competing interests. Training an AE for reconstruction purposes would not necessarily highlight the importance of the input features. On the other hand, training a network to learn which features are important will have further difficulties in reconstructing the whole feature space. Therefore, in contrast to other works in this literature, we tackle both these goals independently.

Pre-training: Learning Latent Representations During pre-training, the main goal is to obtain high-performance reconstruction. This is especially a cumbersome task for high-dimensional data. Accordingly, a deep AE is built. The AE consists of: the Encoder $f(\cdot)$, and the Decoder $g(\cdot)$. Algorithm 1 details the pre-training process, in which the loss function depicted in (1) is minimized. The main purpose of this step is to obtain encodings that hold information portrayed in the original high-dimensional feature space. Minimizing the loss function in (1) guarantees that the lower-

Algorithm 1: Autoencoder Pre-training

Input: training dataset $\mathbf{X} \in \mathbb{R}^{N \times n}$, Encoder network $f(\cdot)$, Decoder network $g(\cdot)$, number of epochs $\mathcal{E}$, learning rate λ, loss function $\mathcal{L}_P$, number of batches $\mathcal{B}$.
Output: Encoder & Decoder parameters $\theta^* = \{\theta^*_{enc}, \theta^*_{dec}\}$.

randomly initialize $\theta = \{\theta_{enc}, \theta_{enc}\}$.
for $e \in \{1...\mathcal{E}\}$ **do**
  for $b \in \{1...\mathcal{B}\}$ **do**
    Compute $\hat{\mathbf{X}}_b = g(f(\mathbf{X}_b, \theta_{enc}), \theta_{dec})$.
    Calculate the loss as $\mathcal{L}_P = \|\mathbf{X}_b - \hat{\mathbf{X}}_b\|_F^2$
    Update $\theta \leftarrow \theta + \nabla_\theta \mathcal{L}_P$
  end for
end for
return $\theta^*_{enc} \leftarrow \theta_{enc}, \theta^*_{dec} \leftarrow \theta_{dec}$

dimensional encodings are globally representative of the original feature space.

Feature Selection using Transfer Learning During the Feature Selection process, a Deep Neural Network (DNN) is built consisting of a Selective Layer and a Deep Encoder. The Selective Layer is a customized multiplicative one-to-one layer that holds weights associated with the input features. Let $\mathbf{w} \in \mathbb{R}^n$ denote the vector of weights w_i associated with input feature X_i for $i \in [1, .., n]$, such that the output of the multiplicative layer, $\mathbf{U} \in \mathbb{R}^{N \times n}$, can be obtained by $\mathbf{XW}$, where $\mathbf{W} = \text{Diag}\{\mathbf{w}\}$ represents a diagonal matrix with the diagonal $\mathbf{w}$. The Selective Layer's weights are required to be non-negative since they depict the relevancy of the features. Also, the non-negativity constraint allows their interpretation to be more meaningful (Xu et al. 2021). The goal is to train the network to drive the weights associated with redundant features to zero. The main driver of $\mathbf{w}$ to zero is the ℓ_1 norm which induces sparsity. After training, the Selective Layer's parameter $\tilde{\mathbf{w}}$ is learned, which is a vector containing zeros (those associated with redundant/unnecessary features) and non-zeros (those associated with surviving/relevant features). Unlike other methods AE-based SOTA methods (Doquet and Sebag 2020; Wu and Cheng 2021), which use the learned associated weights to the input as a direct ranking of importance, we believe the weights of the Selective Layer may be deceiving by collaborating with other parts of the neural network. Thus, the non-zero weights do not in any way portray the level of importance of a feature. In other words, a feature is either relevant or not, but the weights do not in fact represent the importance; i.e. if it is 0, it is irrelevant, and if it is not zero, it is relevant.

The Encoder structure is inherited via Transfer Learning from the pre-trained network, along with the parameters θ^*_{enc} which construct well-representative encodings. The purpose of the Feature Selection process is to reduce features while maintaining the model's reconstruction skill. This is because the encodings learned in the pre-trained network, $\hat{\mathbf{V}} = f(\mathbf{X}, \theta^*_{enc})$, are adequate lower-dimensional representations of the whole original feature space. Usually, in AE applications, we are not only interested in reconstructing the input data but rather learning encodings that take on useful properties from the input space (Goodfellow, Bengio, and Courville 2016). With the focus being on reconstructing the latent representations from a subset of the feature space, the Decoder is not required in the network, which in turn will cut down training time.

Another motivation for neglecting the Decoder in the Feature Selection process is the fact that the encodings can be represented by the following:

$$\mathbf{XA} = \hat{\mathbf{V}} \tag{3}$$

where $\mathbf{A}$ is a nonlinear $n \times p$ transformation matrix, with knowledge of $n > p$. Given $\mathbf{A}$ is a non-square matrix, it is difficult to find an inverse transformation that allows us to obtain $\mathbf{X}$ given $\hat{\mathbf{V}}$. Accordingly, if both the Encoder and the Decoder structures are inherited from the pre-trained network, there is a good chance that different encodings ($\mathbf{V} \neq \hat{\mathbf{V}}$) would be generated. This is specifically because, in the Feature Selection process, the network is trying to force some elements of $\tilde{\mathbf{w}}$ to zeroes, which might force the network to over-fit while reconstructing with some zeroed features.

Accordingly, the target output of Feature Selection's network is the encodings predicted from the pre-trained network. Thus, to learn the model's parameters, we minimize the following objective:

$$\min_{\{\mathbf{w}, \theta_{enc}\}} \|\hat{\mathbf{V}} - f(\mathbf{XW}, \theta_{enc})\|_{\text{F}}^2 + \gamma_1 \|\theta_{enc}\|_1 + \gamma_2 \|\mathbf{w}\|_1 \tag{4}$$

such that $\mathbf{W} \geq 0$ represents the weights' nonnegativity constraint, $\gamma_1 > 0$ represents the coefficient parameter of the Lasso regularization term for layers in the Encoder, $\gamma_2 > 0$ represents the coefficient parameter of the Lasso regularization term for the Selective Layer, $\hat{\mathbf{V}} = f(\mathbf{X}, \theta^*_{enc})$ is the reconstructed encodings in the pre-trained network, and $\tilde{\mathbf{V}} = f(\mathbf{X}\tilde{\mathbf{W}}, \tilde{\theta}_{enc})$ is the reconstructed encodings. As can be seen in (4), additional ℓ_1-norm regularization is applied on the Encoder's layers. That is to avoid the high possibility of the weights exploding in the Encoder's layers, and thus, deceitfully rendering a zeroed or near-zeroed feature as relevant.

Algorithm 2 portrays the training process during Feature Selection.

Selective Deep Autoencoder (SDAE): Inference

Post-training, we combine the findings from both trained networks to produce a network that is capable of reconstructing the whole feature space $\mathbf{X}$ from $\mathbf{X}_{\mathcal{F}_k}$. It should be noted that $0 < |\mathbf{X}_{\mathcal{F}_k}| \leq |\mathbf{X}|$ (where $|\cdot|$ is an operator that refers to the number of non-zero features). This means that, first, the number of important features is not zero, i.e. $\tilde{\mathbf{w}} \neq \vec{0}$. Second, the number of selected features cannot be greater than n; the network cannot add extra features that do not exist and deem them important. It should be further noted that this framework does not choose the number of selected features, k, a-priori to training; k is also consequently learned from training.

SDAE is used as a stand-alone imputation network that takes in as input the selected features and constructs the

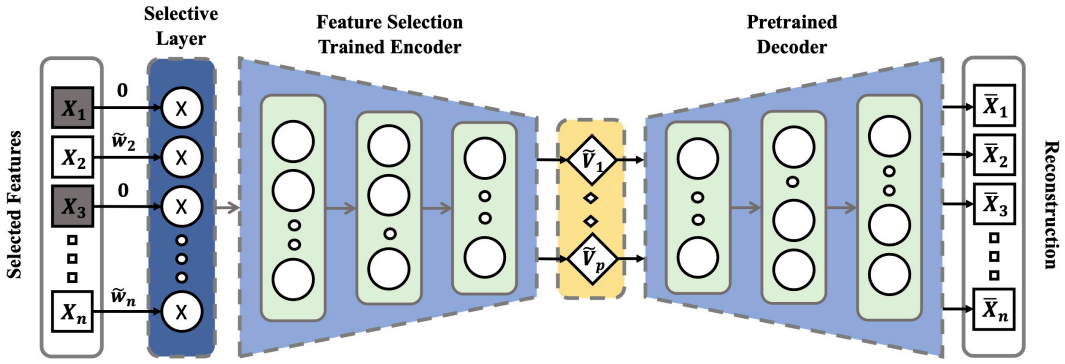

Figure 2: SDAE Inference Network.

Algorithm 2: Feature Selection Training

Input: training dataset $\mathbf{X} \in \mathbb{R}^{N \times n}$, pretrained Encoder network $f(\cdot, \theta^*_{enc})$, number of epochs $\mathcal{E}$, learning rate λ, loss function $\mathcal{L}_{FS}$, number of batches $\mathcal{B}$, lasso regularization terms for Encoder layers γ_1 and Selective Layer γ_2.

Output: Encoder parameters $\tilde{\theta}_{enc}$, Selective Layer parameters $\tilde{\mathbf{w}}$.

initialize $\mathbf{w} \in \mathbb{R}^n, \text{s.t.} \mathbf{w} > 0$.
transfer learning: $\theta_{enc} \leftarrow \theta^*_{enc}$.
obtain $\hat{\mathbf{V}} = f(\mathbf{X}, \theta^*_{enc})$.
for $e \in \{1...\mathcal{E}\}$ **do**
  for $b \in \{1...\mathcal{B}\}$ **do**
    Compute $\tilde{\mathbf{V}}_b = f(\mathbf{X}_b \mathbf{W}, \theta_{enc})$.
    Calculate reconstruction loss $\mathcal{L}_{rec} = \|\hat{\mathbf{V}}_b - \tilde{\mathbf{V}}_b\|^2_F$
    Calculate Encoder layers loss $\mathcal{L}_{enc} = \|\theta_{enc}\|_1$
    Calculate Selective Layer loss $\mathcal{L}_{SL} = \|\mathbf{w}\|_1$
    Update $\mathbf{w}, \theta_{enc} \leftarrow \mathbf{w}, \theta_{enc} + \nabla_{\mathbf{w}, \theta_{enc}}(\mathcal{L}_P + \gamma_1 \mathcal{L}_{enc} + \gamma_2 \mathcal{L}_{SL})$
  end for
end for
return $\tilde{\theta}_{enc} \leftarrow \theta_{enc}, \tilde{\mathbf{w}} \leftarrow \mathbf{w}$

whole feature space, as shown in Figure 2. Performance evaluations are based on the SDAE's outputs. Its weights are set as per the following: $\mathbf{w} = \tilde{\mathbf{w}}$, $\theta_{enc} = \tilde{\theta}_{enc}$, and $\theta_{dec} = \theta^*_{dec}$.

Experimental Study

In this section, we carry out experiments based on simulated data to compare the performance of the SDAE with the FAE (Wu and Cheng 2021), which reports superior performance over other SOTA competing methods for unsupervised feature selection.

Simulated Dataset

The purpose of the simulated dataset is to show how SDAE performs when the features are linearly and/or nonlinearly correlated. Table 1 introduces the features synthesized for this dataset. $\text{Rot}^{\circ}_{\overrightarrow{ab}}(a, 80^{\circ})$ is the resultant transformation of a as a result of a 80° rotation of point vector $\overrightarrow{ab}$ around the global plane axes. On the other hand, $\text{Trans}(a, 1)$ is the translation transformation of a by a unit of 1. Accordingly, features 5 to 16 are linearly correlated to features 1 to 4, and features 17 to 28 are diverse nonlinear mappings of features 1 to 4.

Design of Experiments

There are two experiments undergone for this dataset: first, taking only the linearly correlated features (features 1 to 16), and, second, taking all features. The simulated dataset consists of $N = 6284$ samples. We randomly split it into training and testing sets by a ratio of $80 : 20$. Due to the diverse ranges of each feature, appropriate pre-processing is established for all features; in detail, a standard scalarization was performed for each, for which the mean is removed and scaled to unit variance.

In experiments of SDAE, we set the maximum number of epochs to be 600. We initialize the weights of the Selective Layer by sampling uniformly from $U[0.999999, 0.9999999]$ and the other layers with the Xavier normal initializer. We adopt the Adam optimizer (Kingma and Ba 2015) with an initialized learning rate of 0.001 with a batch size of 20. The number of hidden layers for each of the Encoder and Decoder is set to 3 with each layer holding 12 nodes. The bottleneck layer holds 6 nodes. We only use the linear activation function in SDAE for simplicity. The deep linear version of SDAE can already achieve superior performance.

The experiments' outcomes are evaluated by the reconstruction performance. In specific, we evaluate the mean squared error of the reconstruction done by SDAE and FAE given input of select features. As seen in Table 2, we show for each model the mean squared error of the reconstructions for scaled and real data. The latter corresponds to transforming the predictions, which are in standard scalar form, back to their original scale.

For the hyper-parameters γ_1 and γ_2, we performed 5-fold cross-validation on training dataset across a set of values, and we consequently determined $\gamma_1 = 1e-5$ and $\gamma_2 = 2e-3$ for experiment with linear features only, and $\gamma_1 = 1e-5$ and $\gamma_2 = 1e-2$ for experiment with all features included. Since γ_2 is the obvious driver of the features to zero, we aim to obtain the largest possible γ_2 that guarantees

#	Feature	Description
1	x	$\frac{20\pi}{N-1}t - 10\pi,\ \forall t = \{0, ..., N-1\}$
2	y	$x + \epsilon_y, \quad \text{s.t. } \epsilon_y \sim \mathcal{N}(0, 0.1)$
3	u	$x + \epsilon_u, \quad \text{s.t. } \epsilon_u \sim \mathcal{N}(0, 0.1)$
4	v	$x + \epsilon_v, \quad \text{s.t. } \epsilon_v \sim \mathcal{N}(0, 0.1)$
5	$\text{Rot}_{\overrightarrow{xy}}(x, 80°)$	$x \cdot cos(80°) - y \cdot sin(80°)$
6	$\text{Rot}_{\overrightarrow{xy}}(y, 80°)$	$x \cdot sin(80°) + y \cdot cos(80°)$
7	$\text{Rot}_{\overrightarrow{uv}}(u, 80°)$	$u \cdot cos(80°) - v \cdot sin(80°)$
8	$\text{Rot}_{\overrightarrow{uv}}(v, 80°)$	$u \cdot sin(80°) + v \cdot cos(80°)$
9	$\text{Trans}(x, 1)$	$x' = x + 1$
10	$\text{Trans}(y, 1)$	$y' = y + 1$
11	$\text{Trans}(u, 1)$	$u' = u + 1$
12	$\text{Trans}(v, 1)$	$v' = v + 1$
13	$\text{Rot}_{\overrightarrow{x'y'}}(x', 80°)$	$x' \cdot cos(80°) - y' \cdot sin(80°)$
14	$\text{Rot}_{\overrightarrow{x'y'}}(y', 80°)$	$x' \cdot sin(80°) + y' \cdot cos(80°)$
15	$\text{Rot}_{\overrightarrow{u'v'}}(u', 80°)$	$u' \cdot cos(80°) - v' \cdot sin(80°)$
16	$\text{Rot}_{\overrightarrow{u'v'}}(v', 80°)$	$u' \cdot sin(80°) + v' \cdot cos(80°)$
17	$\text{Cos}(x)$	$cos(x)$
18	$\text{Cos}(y)$	$cos(y)$
19	$\text{Cos}(u)$	$cos(u)$
20	$\text{Cos}(v)$	$cos(v)$
21	$\text{Sin}(x)$	$sin(x)$
22	$\text{Sin}(y)$	$sin(y)$
23	$\text{Sin}(u)$	$sin(u)$
24	$\text{Sin}(v)$	$sin(v)$
25	x^2	x^2
26	y^2	y^2
27	u^2	u^2
28	v^2	v^2

Table 1: Simulated Dataset Description.

the smallest possible selection of features while still having very low reconstruction error.

All experiments are done with Python 3.9.13, Tensorflow 2.10.0, Keras 2.10.0, and Scikit-Learn 1.3.0. The codes can be found at https://github.com/irda-lab/SDAE.

Results

The testing set performance comparison, in terms of scaled and reconstruction mean squared errors, between SDAE and FAE is shown in Table 2. In specific, results corresponding to two simulation case studies are shown: first, with only linear features (i.e. features 1 to 16), and second, all features (i.e. features 1 to 28).

For the first simulation case study in Table 2 involving linear features, the optimal SDAE selects only one feature. This makes sense since all 1 to 16 features are highly linearly correlated to feature x as shown in Table 1. SDAE automatically learns that the optimal number of features selected is $k = 1$. However, for FAE, we attempt different k values and then learn that $k = 1$ is optimal. Note, that the linear features were constructed in such a way that $k = 1$ is optimal and we wanted to see if SDAE can automatically learn it. SDAE slightly outperforms FAE when its selected feature count is $k = 2$, but otherwise offers comparable results.

For the second simulation case study in Table 2 where all features were involved in the model, the optimal SDAE selects 4 features. Accordingly, we pick for FAE $k = 3, 4, 5$ which are comparable to the number of features selected by SDAE. SDAE clearly outperforms FAE when the number of features selected is equivalent with a real reconstruction mean squared error of $2.32\text{e}{-3}$ to 0.9951. Even for an additional feature in the model for FAE, represented by $k = 5$, the model's performance is considered very far from that of SDAE's. This is specifically attributed to SDAE's main quality in extracting quality information in the encodings after going through deep layers in the encoder. FAE, and other SOTA AE-based unsupervised feature selection methods (Abid, Balin, and Zou 2019; Doquet and Sebag 2020), rely on a shallow architecture to extract information. This has proven to be difficult when there are nonlinear features. The penultimate row in Table 2 shows the real reconstruction mean squared error of the nonlinear features (features 17 to 28 in Table 1). Again, SDAE shows that it is capable of reconstructing these nonlinear features that the FAE finds hard to reconstruct. Moreover, for $k = 3$, FAE converges at an extremely bad reconstruction error. This shows that selecting at least four features serves the purpose of reconstruction.

Benchmark Comparisons

In this section, we perform experiments to extensively assess SDAE by comparing it with contemporary methods on several benchmarking datasets:

- **Mice Protein** (Higuera, Gardiner, and Cios 2015) dataset includes measurements of protein expression levels in the cortex of both normal and trisomic mice. Dataset size = $(1080, 561)$.
- **ISOLET** (Cole and Fanty 1994) comprises preprocessed speech recordings where individuals speak the names of English alphabet letters. Dataset size = $(7797, 617)$.
- **MNIST** (LeCun et al. 1998) contains grayscale images of hand-written digits with each image being 28-by-28 pixels in size. Dataset size = $(10000, 784)$.

Design of Experiments

MNIST's original dataset has 60000 training and 10000 testing images. Following (Abid, Balin, and Zou 2019), we randomly choose 6000 samples from the training set to train and validate and 4000 from the testing set for testing. For the other two datasets, we randomly split them into training, and testing sets by a ratio of $80 : 20$.

In experiments of SDAE, we follow some of the previous sections' parameter settings, in terms of the maximum number of epochs, the network's layers initializations, and the optimizer used with the corresponding learning rate. For the hyper-parameter setting, we perform a grid search on the validation set and then choose the optimal ones. The activation function used for all layers is Scaled Exponential Linear Units (selu) (Klambauer et al. 2017), except the reconstruction layer which maintains the linear activation function.

All other methods require specifying a-priori a number of selected features k. In contrast, SDAE intrinsically learns the subset of features necessary for selection while training. For the Mice Protein dataset, $k = 10$, while for the other two datasets, $k = 50$. To make a fair comparison with the

Simulation Case	**Linear Features**				**All Features**			
Models	**FAE**	**FAE**	**FAE**	**SDAE**	**FAE**	**FAE**	**FAE**	**SDAE**
FS parameter	$k = 1$	$k = 2$	$k = 3$	$\gamma_2 = 2\text{e}{-3}$	$k = 3$	$k = 4$	$k = 5$	$\gamma_2 = 1\text{e}{-2}$
Scaled Recons.	$2.82\text{e}{-8}$	$2.06\text{e}{-7}$	$4.15\text{e}{-8}$	$1.85\text{e}{-7}$	0.1498	$1.25\text{e}{-5}$	$5.49\text{e}{-6}$	$3.44\text{e}{-7}$
Real Recons.	$9.08\text{e}{-6}$	$6.73\text{e}{-5}$	$1.36\text{e}{-5}$	$6.04\text{e}{-5}$	12855	0.4283	0.1027	$2.32\text{e}{-3}$
Nonlinear Recons.	-	-	-	-	29994	0.9951	0.2388	$5.21\text{e}{-3}$
# of Selected Feats.	1^*	2^*	3^*	1	3^*	4^*	5^*	4

* Features are a-priori selected using k

Table 2: Performance Results on Simulated Data.

Dataset	**LS**	**AEFS**	**UDFS**	**MCFS**	**AgnoS-S**	**CAE**	**FAE**	**SDAE**	k_{SDAE}
Mice Protein	575 ± 118	132 ± 178	9 ± 6	165 ± 304	38 ± 14	32 ± 1	14 ± 5	26 ± 2	10 ± 0
ISOLET	304 ± 47	104 ± 7	19 ± 1	154 ± 32	35 ± 5	13 ± 0	15 ± 0	16 ± 1	49.9 ± 1.9
MNIST	305 ± 0	67 ± 1	29 ± 2	128 ± 3	55 ± 5	19 ± 0	19 ± 0	19 ± 0	49.2 ± 1.5

Table 3: Linear reconstruction error ($\times 10^{-3}$) with selected features by different algorithms.

Dataset	**LS**	**AEFS**	**UDFS**	**MCFS**	**AgnoS-S**	**CAE**	**FAE**	**SDAE**
Mice Protein	17.3 ± 4.0	13.7 ± 3.6	65.1 ± 4.0	17.4 ± 5.9	63.2 ± 37.2	66.9 ± 4.8	87.8 ± 7.3	94.4 ± 1.4
ISOLET	$11.1 \pm .7$	$3.4 \pm .9$	73.9 ± 5.8	6.0 ± 1.2	36.2 ± 17.0	$87.9 \pm .6$	$89.0 \pm .6$	$88.5 \pm .6$
MNIST	12.4 ± 1.0	11.2 ± 1.0	88.1 ± 1.6	13.0 ± 4.1	43.5 ± 15.6	$92.5 \pm .4$	$92.9 \pm .7$	$91.3 \pm .3$

Table 4: Classification accuracy (%) with selected features by different algorithms. Values for k_{SDAE} are shown in Table 3.

other methods, we tune for each experiment λ_2 of the SDAE loss function to obtain a number of features selected close to k. λ_2 is the regularizer mainly responsible for driving the Selective Layer's weights to zero.

To evaluate the models, certain metrics are used. The first is the linear reconstruction error, measured in MSE; following Abid, Balin, and Zou (2019), after selecting the features, a linear regression model with no regularization is trained to reconstruct the original features, and the consequent linear reconstruction error is recorded. The second metric is classification accuracy, which is measured by passing the selected features to an extremely randomized trees classifier (Geurts, Ernst, and Wehenkel 2006) to benchmark the quality of the selected subset of features. It should be noted that all methods were trained in an unsupervised manner, meaning that the classification targets were not used in training but rather only in training the downstream classifier.

Results on Real Datasets

The experimental results on reconstruction and classification with the selected features by different algorithms are reported in Tables 3 and 4. For all the results, we implement 5 runs with random splits on the fixed dataset to present mean results and standard errors. The results of all other benchmarks are obtained from Wu and Cheng (2021).

From Table 3, it is seen that SDAE performs among the top 3 smallest reconstruction errors among the baseline methods on all three datasets, indicating its strong ability to represent the original data. This is an excellent result given that the average of the selected Features per experiment was slightly lower than k assigned for other methods. In addition to the reconstruction of all original features, SDAE also proved it can allow comparably high performances in downstream classification, in comparison to the seven other SOTA algorithms, as shown in Table 4. However, in one of the datasets, i.e. Mice Protein, where the downstream classification proved to be tough for all other baselines, with the top two amongst them being CAE (66.9%) and FAE (87.8%), SDAE was capable of achieving a superior accuracy of 94.4%. This shows the prowess of SDAE's deeper architecture in disentangling nonlinear inter-feature correlation to select the most meaningful subset of features.

Conclusion

In this paper, we propose a novel framework, SDAE, for the purpose of unsupervised feature selection. It is a deep AE-based learning method that contains a customized Selective Layer, which intrinsically learns a subset of features that is well-representative of the whole feature space. Through reconstruction learning, the network learns globally representative lower-dimensional deep encodings, which are leveraged to aid in selecting relevant features capable of generalizing the whole feature space. Furthermore, SDAE's deep structure allows compression of quality information in the encodings. This is particularly shown in its superior performance in a nonlinearly correlated simulated dataset. Moreover, SDAE was compared to other baseline methods evaluated on three real datasets. It was shown to outperform the other methods not only in reconstruction but also in a downstream classification task.

References

Abid, A.; Balin, M. F.; and Zou, J. 2019. Concrete Autoencoders: Differentiable Feature Selection and Reconstruction. In *Proceedings of the 36th International Conference on Machine Learning, PMLR*, 444–453.

Alelyani, S.; Tang, J.; and Liu, H. 2018. Feature Selection for Clustering: A Review. In *Data Clustering*, 29–60. Chapman and Hall/CRC.

Bolón-Canedo, V.; and Remeseiro, B. 2020. Feature selection in image analysis: a survey. *Artificial Intelligence Review*, 53: 2905–2931.

Cai, D.; Zhang, C.; and He, X. 2010. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*, 333. ACM Press. ISBN 9781450300551.

Chandra, B.; and Sharma, R. K. 2015. Exploring autoencoders for unsupervised feature selection. In *2015 International Joint Conference on Neural Networks (IJCNN)*, volume 2015-September, 1–6. IEEE. ISBN 978-1-4799-1960-4.

Cole, R.; and Fanty, M. 1994. ISOLET. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C51G69.

Dièn, B.; Rodrigues, J. J.; Diallo, O.; Ndoye, E. H. M.; and Korotaev, V. V. 2020. Data management techniques for Internet of Things. *Mechanical Systems and Signal Processing*, 138: 106564.

Doquet, G.; and Sebag, M. 2020. Agnostic Feature Selection. In *Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2019*, 343–358. ISBN 978-3-030-46150-8.

Doshi-Velez, F.; and Kim, B. 2017. Towards A Rigorous Science of Interpretable Machine Learning. arXiv:1702.08608.

Geurts, P.; Ernst, D.; and Wehenkel, L. 2006. Extremely randomized trees. *Machine Learning*, 63: 3–42.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. Autoencoders. In *Deep Learning*, 499–523. MIT Press.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and Harnessing Adversarial Examples. *International Conference on Learning Representations*.

Guyon, I.; and Elisseeff, A. 2003. An Introduction to Variable and Feature Selection. *The Journal of Machine Learning Research*, 3: 1157–1182.

Hamo, Y.; and Markovitch, S. 2005. The COMPSET Algorithm for Subset Selection. In *Proceedings of the 19th international joint conference on Artificial intelligence, IJCAI'05*.

Han, K.; Wang, Y.; Zhang, C.; Li, C.; and Xu, C. 2018. Autoencoder Inspired Unsupervised Feature Selection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2941–2945. IEEE. ISBN 978-1-5386-4658-8.

He, X.; Cai, D.; and Niyogi, P. 2005. Laplacian Score for Feature Selection. In Weiss, Y.; Schölkopf, B.; and Platt, J., eds., *Advances in Neural Information Processing Systems*, volume 18. MIT Press.

Higuera, C.; Gardiner, K.; and Cios, K. 2015. Mice Protein Expression. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C50S3Z.

Hinton, G. E.; and Salakhutdinov, R. R. 2006. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313: 504–507.

Hotelling, H. 1933. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24: 417–441.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *The 3rd International Conference for Learning Representations*.

Klambauer, G.; Unterthiner, T.; Mayr, A.; and Hochreiter, S. 2017. Self-Normalizing Neural Networks. In *The 31st International Conference on Neural Information Processing Systems*, 972–981. Curran Associates Inc.

Kramer, M. A. 1991. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37: 233–243.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86: 2278–2323.

Li, J.; Cheng, K.; Wang, S.; Morstatter, F.; Trevino, R. P.; Tang, J.; and Liu, H. 2018. Feature Selection. *ACM Computing Surveys*, 50: 1–45.

Li, Y.; Chen, C.-Y.; and Wasserman, W. W. 2016. Deep Feature Selection: Theory and Application to Identify Enhancers and Promoters. *Journal of Computational Biology*, 23: 322–336.

Pinaya, W. H. L.; Vieira, S.; Garcia-Dias, R.; and Mechelli, A. 2020. Autoencoders. In *Machine Learning*, 193–208. Elsevier.

Ramjee, S.; and Gamal, A. E. 2020. Efficient Wrapper Feature Selection using Autoencoder and Model Based Elimination. arXiv:1905.11592.

Robnik-Šikonja, M.; and Kononenko, I. 2003. Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53: 23–69.

Savargaonkar, M.; Oyewole, I.; Chehade, A.; and Hussein, A. A. 2022. Uncorrelated Sparse Autoencoder With Long Short-Term Memory for State-of-Charge Estimations in Lithium-Ion Battery Cells. *IEEE Transactions on Automation Science and Engineering*, 1–12.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P.-A. 2010. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, 11: 3371–3408.

Wang, G.; Song, Q.; Sun, H.; Zhang, X.; Xu, B.; and Zhou, Y. 2013. A Feature Subset Selection Algorithm Automatic Recommendation Method. *Journal of Artificial Intelligence Research*, 47: 1–34.

Wu, X.; and Cheng, Q. 2021. Fractal Autoencoders for Feature Selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35: 10370–10378.

Xu, J.; Yu, M.; Shao, L.; Zuo, W.; Meng, D.; Zhang, L.; and Zhang, D. 2021. Scaled Simplex Representation for Subspace Clustering. *IEEE Transactions on Cybernetics*, 51: 1493–1505.

Xu, X.; Gu, H.; Wang, Y.; Wang, J.; and Qin, P. 2019. Autoencoder Based Feature Selection Method for Classification of Anticancer Drug Response. *Frontiers in Genetics*, 10.

Yang, Y.; Shen, H. T.; Ma, Z.; Huang, Z.; and Zhou, X. 2011. L2,1-Norm Regularized Discriminative Feature Selection for Unsupervised Learning. In *International Joint Conferences on Artificial Intelligence*.

Yu, L.; and Liu, H. 2003. Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, 856–863. AAAI Press.

Zhao, Z.; and Liu, H. 2007. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th international conference on Machine learning - ICML '07*, 1151–1157. ACM Press. ISBN 9781595937933.