Neural Computing and Applications (2023) 35:21883–21902
https://doi.org/10.1007/s00521-023-08938-7

ORIGINAL ARTICLE

Nonlinear feature selection using sparsity-promoted centroid-encoder

Tomojit Ghosh[1] · Michael Kirby[2]

Received: 22 March 2023 / Accepted: 1 August 2023 / Published online: 22 August 2023

Abstract
The contribution of our work is two-fold. First, we propose a novel feature selection technique, sparsity-promoted centroid-encoder (SCE). The model uses the nonlinear mapping of artificial neural networks to reconstruct a sample as its class centroid and, at the same time, apply a ℓ_1-penalty to the weights of a sparsity promoting layer, placed between the input and first hidden layer, to select discriminative features from input data. Using the proposed method, we designed a feature selection framework that first ranks each feature and then, compiles the optimal set using validation samples. The second part of our study investigates the role of stochastic optimization, such as Adam, in minimizing ℓ_1-norm. The empirical analysis shows that the hyper-parameters of Adam (mini-batch size, learning rate, etc.) play a crucial role in promoting feature sparsity by SCE. We apply our technique to numerous real-world data sets and find that it significantly outperforms other state-of-the-art methods, including LassoNet, stochastic gates (STG), feature selection networks (FsNet), supervised concrete autoencoder (CAE), deep feature selection (DFS), and random forest (RF).

Keywords Feature selection · Nonlinear feature selection · Sparsity-promoted centroid-encoder · Centroid-encoder · ℓ_1-norm · Sparse optimization · Feature sparsity · Neural networks

1 Introduction

High-dimensional data sets are now ubiquitous in Machine Learning workflows for data driven knowledge discovery. For example, in bioinformatics, the researchers seek to understand the gene expression level with microarray or next-generation sequencing techniques where each point consists of over 50,000 measurements [1–4]. The abundance of features demands the development of feature selection algorithms to improve Machine Learning explainability in classification problems. For high-dimensional data samples, it is often typical that good classification rates may be due to a small fraction of the measured features. Hence, selecting the discriminatory features from large feature sets are essential to understanding the underlying data as well as explaining, interpreting and trusting predictive models. For example, the discovery of drug therapies revolves around the identification of biomarkers that characterize the biological processes associated with the host immune response to infection by respiratory viruses such as influenza [5]. Additional benefits of feature selection include improved visualization and understanding of data, reducing storage requirements, and faster algorithm training times.

Feature selection can be accomplished in various ways that can be broadly categorized into the filter, wrapper, and embedded methods. In a filter method, each variable is ordered based on a score. After that, a threshold is used to select the relevant features [6]. Variables are usually ranked using correlation [7, 8] and mutual information [9, 10]. In contrast, a wrapper method uses a model and determines the importance of a feature or a group of features by the generalization performance of the predetermined model [11, 12]. Since evaluating every possible combination of features becomes an NP-hard problem, heuristics are used to find a subset of features. Wrapper methods are computationally intensive for larger data sets,

✉ Tomojit Ghosh
Tomojit-Ghosh@utc.edu; Tomojit.Ghosh@colostate.edu

Michael Kirby
Kirby@math.colostate.edu

[1] Department of Computer Science and Engineering, University of Tennessee, 615 McCallie Avenue, Chattanooga, TN 37403, USA

[2] Department of Mathematics, Colorado State University, 841 Oval Drive, Fort Collins, CO 80523, USA

in which case search techniques like Genetic Algorithm (GA) [13] or Particle Swarm Optimization (PSO) [14] are used. In embedded methods, feature selection criteria are incorporated within the model, i.e., the variables are picked during the training process [15]. Iterative Feature Removal (IFR) uses the absolute weight ratios of a Sparse SVM model as a criterion to extract features from the high dimensional biological data set [5].

Mathematically feature selection problem can be posed as an optimization problem on ℓ_0-norm, i.e., how many predictors are required for a machine learning task. As the minimization of ℓ_0 is intractable (non-convex and non-differentiable), ℓ_1-norm is used instead, which is a convex proxy of ℓ_0 [16]. Note ℓ_1 is not differentiable at 0, but the problem can be tackled by leveraging on sub-gradient methods [17, 18]. Since the introduction of these seminal papers [19, 20], the use of 1-norm is now widespread. For example, the ℓ_1 has been used for the feature selection task in linear [5, 21–24] as well as in nonlinear regime [25–27]. Popularity notwithstanding, note that it is only a proxy for sparsity, and can generate small weights (the shrinkage problem) [28, 29], and has the potential non-unique solutions [30].

This paper proposes a new embedded variable selection approach called Sparsity-promoted Centroid-Encoder (SCE) to extract features when class labels are available. Our method extends the Centroid-Encoder model [31, 32], where we applied a ℓ_1-penalty to a sparsity promoting layer between the input and the first hidden layer. We evaluate this proposed model SCE on diverse data sets and show that the selected features produce better generalization than other state-of-the-art techniques. As a feature selection tool, SCE uses a single model for the multi-class problem without the need to create multiple one-against-one binary models typical of linear methods, e.g., Lasso [16], or Sparse SVM [24]. The work of [25] also uses a similar sparse layer between the input and the first hidden with an Elastic net penalty while minimizing the classification error with a softmax layer. The authors used Theano's symbolic differentiation [33] to impose sparsity. In contrast, our approach minimizes the Centroid-Encoder loss with an explicit differentiation of the ℓ_1 function using the sub-gradient. Unlike DFS, our model can capture the intra-class variability by using multiple centroids per class. This property is beneficial for multi-modal data sets.

1.1 Summary of novelty

Here, we summarize the novel aspects of our work. The performance implications of these innovations will be explored via direct comparison with many of state-of-the-art algorithms in the context of benchmark data sets from the literature.

- We propose a novel nonlinear feature selection technique, called Sparsity-promoted Centroid-Encoder (SCE).
- SCE minimizes the distortion error of each class in the ambient space and, at the same time, uses a ℓ_1-penalty on the sparse layer to discard features from input not essential to reconstruct the class centroids.
- One key attribute of SCE is that it can extract informative features by capturing the intra-class variance using multiple centroids per class. This property of SCE distinguishes itself from other neural network-based feature selection techniques, such as LassoNet [34], Concrete Autoencoders [35], FsNet [36], Stochastic Gate [37], which do not model the multi-modal nature of data (data sets whose classes appear to have multiple clusters) during feature selection.
- SCE requires the solution of a non-convex optimization problem. Training such models has the potential to produce a non-unique set of selected features as a consequence of local minima. We propose a framework to select the most robust features to address this limitation of all non-convex feature selection algorithms.
- We also address the challenges of minimizing ℓ_1-norm using stochastic optimization. We empirically show that the hyper-parameters associated with stochastic optimization, such as learning rate and mini-batch size, play a critical role in promoting sparsity using SCE.

The article is organized as follows: In Sect. 2, we review related work, for both linear and nonlinear feature selection techniques. In Sect. 3, we present the formulation of Sparsity-promoted Centroid-Encoder (SCE) with analysis to understand the model. In Sect. 4, we present a robust feature selection workflow. Section 5 offers an array of experiments to show the challenges of minimizing ℓ_1-norm using stochastic optimization. In Sect. 6, we apply SCE to a range of bench-marking data sets taken from the literature and compare it with other state-of-the-art methods. Finally, we present a discussion and possible extension of our model in Sect. 7.

2 Related work

Feature selection has a long history spread across many fields, including bioinformatics, document classification, data mining, hyperspectral band selection, computer vision. It is an active research area, and numerous techniques exist to accomplish the task. We describe the literature related to the embedded methods where the selection criteria are part of a model. The model can be either linear or nonlinear.

2.1 Feature selection using linear models

Linear models are widely used in Machine Learning for classification and regression. These models approximate the output as a linear combination of input variables (features), i.e., $y \approx f(x) = w^T x + b$ where w and b are the model parameters. From the optimization perspective, a linear model takes the following form: $\underset{\theta}{\text{minimize}}\ \ l(y, f(x, \theta))$ where l is a loss function and θ is the parameter set. Adding a ℓ_1 penalty on the parameter set θ gives a feature selector, least absolute shrinkage and selection operator or Lasso [16], which takes the form:

$$\underset{\theta}{\text{minimize}}\ \ l(y, f(x, \theta)) \ + \ \lambda \|\theta\|_1 \tag{1}$$

where λ is a hyper parameter which controls the sparsity of the model. Since its inception, the model has been used extensively for feature selection on various data sets [21–23]. Elastic net, proposed by Zou et al. [29], combined the Lasso penalty with the Ridge Regression penalty [38] to overcome some limitations of Lasso. The Elastic net is defined as following:

$$\underset{\theta}{\text{minimize}}\ \ l(y, f(x, \theta)) \ + \ (1 - \alpha)\|\theta\|_1 \ + \ \alpha \|\theta\|_2^2 \tag{2}$$

where $\alpha \in [0, 1)$ and the term $(1 - \alpha)\|\theta\|_1 + \alpha\|\theta\|_2^2$ known as elastic net penalty. Elastic net has been widely applied, e.g., [39–41]. Note both Lasso and Elastic net are convex in the parameter space. Also see the following works which address the issues with Lasso [42–46].

Support Vector Machines (SVM) [47] is a state-of-the-art model for classification, regression and feature selection. SVM-RFE is a linear feature selection model which iteratively removes the least discriminative features until a parsimonious set of predictive features are selected [48]. Arbitrary p-norm separating hyperplanes were proposed by [49]. IFR [5], on the other hand, selects a group of discriminatory features at each iteration and eliminates them from the data set. The process repeats until the accuracy of the model starts to drop significantly. Note IFR uses Sparse SVM (SSVM), which minimizes the l_1 norm of the model parameters. Lasso, Elastic Net, and SVM-based techniques are primarily suitable for binary problems, i.e., a single model cannot handle multiple classes. These models are extended to the multi-class regime by combining several binary one-against-one (OAO) or one-against-all (OAA) models. For example, [24] used 120 Sparse SVM models to select discriminative bands from the Indian Pine data set, which has 16 classes.

On the other hand, Random forest (RF) [50], a decision tree-based technique, finds features from multi-class data using a single model. The model does not use Lasso or Elastic net penalty for feature selection. Instead, the model weighs the importance of each feature by measuring the out-of-bag error. Warda et al. proposed a hybrid feature selection technique (HBAPSO) for breast cancer prediction [51]. The algorithm uses a combination of particle swarm optimization (PSO) [14] and bat algorithm (BA) [52] for feature pruning. The work of Dai et al. uses label correlation and instance correlation with the $\ell_{2,1}$-norm to extract features from multi-label data (CMFSS) [53]. More information on feature selection on high dimensional microarray data can be found [54].

2.2 Feature selection using deep neural networks

While the linear models are fast and convex, they do not capture the nonlinear relationship among the input features (unless a kernel trick is applied). Because of the shallow architecture, these models do not learn a high-level representation of input features. Moreover, there is no natural way to incorporate multi-class data in a single model. Nonlinear models based on deep neural networks overcome these limitations. In this section, we will briefly discuss a handful of such models.

Group Lasso [55] was modified to impose sparsity on a group of variables instead of a single variable [26]. They applied the group sparsity simultaneously on the input and the hidden layers to remove features from the input data and the hidden activation. On MNIST, their algorithm discarded more than 200 features from the input vector with an accuracy of 97% on the test data. Although on the Forest Cover data set, the algorithm used most of the input variables, 52.7 on average out of 54. Deep feature selection (DFS), a multilayer neural network-based feature selection technique, was proposed by [25]. As a sparse regularization, the authors used elastic-net [29] on the variables of the feature selection layer to induce sparsity. The standard soft-max function is used in the output layer for classification. With this setup, the network is trained in an end-to-end fashion by error backpropagation. Despite the deep architecture, its accuracy is not competitive, and experimental results have shown that the method did not outperform the random forest (RF) method. Kim et al. proposed a heuristics-based technique [56], EP-DNN, to assign importance to each feature. Unlike [25, 26], EP-DNN does not use sparsity or group sparsity during training; instead, the importance of a feature is calculated using a backpropagation-like technique after the training step is done, making it a multi-step process. The authors evaluated the model with only one biological data set. On the other hand, Roy et al. proposed to use ReLU activation to measure the contribution of an input feature toward hidden activation of the next layer [57]. The approach makes the

feature selection and training of the deep network in a single step. Unlike the supervised methods [25, 26, 56, 57], Han et al. developed an unsupervised feature selection technique based on the autoencoder architecture [58]. Applying a $l_{2,1}$-penalty to the weights coming out from each input node, the authors measure the contribution of each feature while reconstructing the input. The model removes the input features with a minimum contribution to sample reconstruction. Similar to the previous work, Taherkhani et al. proposed [59] an RBM[60]-based feature selection model which discards a feature if the reconstruction error does not increase after setting the corresponding input to zero.

Recently, Balin et al. proposed an end-to-end unsupervised feature selection technique, namely Concrete Autoencoders (CAE) [35]. The authors utilize the concrete random variable, a continuous approximation of a one-hot vector in the feature selection layer. One of the attractive features of CAE is that its cost function is differentiable, and the model picks a subset of original features by gradually minimizing the temperature of the concrete feature selector layer using an annealing scheme. Note CAE requires the user to specify the number of features to be selected from the data. Stochastic Gates, proposed by [37] incorporates continuous relaxation of the Bernoulli distribution to approximate ℓ_0-norm. Like CAE, the cost of the Stochastic Gate is also differentiable, and the model assumes the input features follow a Gaussian distribution. The FsNet, proposed by Singh et al. [36], designed for high-dimensional biological data sets, also uses a Concrete feature selection layer along with Diet Networks [61] to reduce the model size. LassoNet [34], on the other hand, uses a skip connection to measure the contribution of a feature using the Lasso penalty and only allows a feature to participate in hidden units if it is still active. Uzma et al. proposed Gene encoder [62] which is an unsupervised feature selection technique based on deep architecture. We left the brief description of these models in (Table 1).

3 Sparsity-promoted centroid-encoder

Centroid-Encoder (CE) neural networks are the starting point of our approach [31, 32, 63]. We present a brief overview of CEs and demonstrate how they can be extended to perform nonlinear feature selection.

3.1 Centroid-encoder

The CE neural network is a variation of an autoencoder and can be used for both visualization and classification tasks. Consider a data set with N samples and M classes. The classes denoted $C_j, j = 1, \ldots, M$ where the indices of the data associated with class C_j are denoted I_j. We define centroid of each class as $c_j = \frac{1}{|C_j|}\sum_{i\in I_j} x^i$ where $|C_j|$ is the cardinality of class C_j. Unlike autoencoder, which maps each point x^i to itself, the CE maps each point x^i to its class centroid c_j by minimizing the following cost function over the parameter set θ:

$$\mathcal{L}_{ce}(\theta) = \frac{1}{2N}\sum_{j=1}^{M}\sum_{i\in I_j} \|c_j - f(x^i;\theta))\|_2^2 \tag{3}$$

The mapping f is composed of a dimension reducing mapping g (encoder) followed by a dimension increasing reconstruction mapping h (decoder). The output of the encoder is used as a supervised visualization tool [31, 32] and attaching another layer to map to the one-hot encoded labels performs robust classification [63].

3.2 Sparsity-promoted centroid-encoder for feature selection

The Sparsity-promoted Centroid-Encoder (SCE) is a modification to the centroid-encoder architecture as shown in Fig. 1. Unlike centroid-encoder, we have not used a bottleneck architecture as visualization is not our aim here. The input layer is connected to the first hidden layer via the sparsity promoting layer (SPL). Each node of the input layer has a weighted one-to-one connection to each node of the SPL. The number of nodes in these two layers is the same. The nodes in SPL do not have any bias or nonlinearity. The SPL is fully connected to the first hidden layer; therefore, the weighted input from the SPL will be passed to the hidden layer in the same way that of a standard feed forward network. During training, a ℓ_1 penalty will be applied to the weights connecting the input layer and SPL layer. The sparsity promoting ℓ_1 penalty will drive most of the weights to near zero, and the corresponding input nodes/features can be discarded. Therefore, the purpose of the SPL is to select important features from the original input. Note we only apply the ℓ_1 penalty to the parameters of the SPL.

Denote θ_{spl} to be the parameters (weights) of the SPL and θ to be the parameters of the rest of the network. The cost function of SCE is given by

$$\mathcal{L}_{sce}(\theta) = \frac{1}{2N}\sum_{j=1}^{M}\sum_{i\in I_j} \|c_j - f(x^i;\theta))\|_2^2 + \lambda\|\theta_{spl}\|_1 \tag{4}$$

where λ is the hyperparameter which controls the sparsity. A larger value of λ will promote higher sparsity resulting more near-zero weights in SPL. In other words, λ is a knob that controls the number of features selected from the input data.

Table 1 Brief description of the feature selection techniques

Model	Description
LASSO [16]	A linear regression/classification model that uses ℓ_1-penalty for variable selection
ElasticNet [29]	A linear regression/classification model that applies both ℓ_1 and ℓ_2-penalty for variable selection
SSVM [5]	A SVM-based linear model which minimizes the ℓ_1-norm for feature sparsity
RF [50]	A decision tree-based model which calculate importance of each feature by measuring the out-of-bag error
HBAPSO [51]	A hybrid feature selection technique using particle swarm (PSO) [14] and bat algorithm (BA) [52]
CMFSS [53]	Using $\ell_{2,1}$-norm, the model picks informative features using the label correlation and instance correlation in a collaborative manner
DFS [25]	A neural network-based model which uses elastic net penalty for feature selection
SG-ANN [26]	This ANN-based feature selection technique use group-sparsity for feature selection
EP-DNN [56]	A multi-step ANN-based model which calculates feature importance using a backpropagation like algorithm
Roy et al. [57]	This ANN-based technique uses ReLU unit for feature selection
Han et al. [58]	An autoencoder-based unsupervised technique. Uses $l_{2,1}$-penalty to calculate each feature's contribution in reconstruction
Taherkhani et al. [59]	An RBM [60]-based unsupervised technique which discards a feature if setting it to 0 doesn't increase reconstruction loss
CAE [35]	A differentiable technique which uses concrete random variable to select an input feature. The feature selection layer is attached to an Autoencoder model, making it unsupervised
STG [37]	A differentiable technique which approximates ℓ_0-norm by continuous relaxation of the Bernoulli distribution
FsNet [36]	The model uses Diet Network [61] along with CAE [35] to select features from high-dimensional biological data
LassoNet [34]	Uses a skip connection to measure the contribution of a feature using the Lasso penalty [16]
Gene encoder [62]	An unsupervised model based on deep neural networks

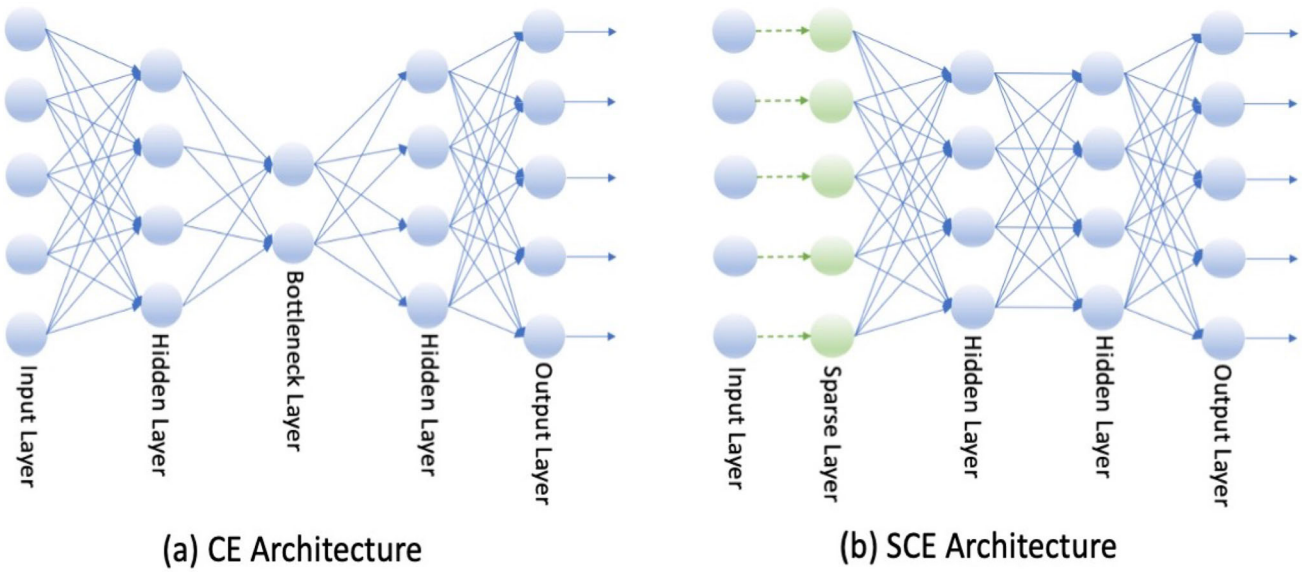

Fig. 1 Architecture of Centroid-Encoder and Sparsity-promoted Centroid-Encoder. Notice the Centroid-Encoder uses a bottleneck architecture which is helpful for visualization. In contrast, the Sparsity-promoted Centroid-Encoder does not use any bottleneck architecture; instead, it employs a sparse layer between the input and the first hidden layer to promote feature sparsity

Like centroid-encoder, we trained sparsity-promoted centroid-encoder using error backpropagation, which requires the gradient of the cost function of Eq. 4. As ℓ_1 function is not differentiable at 0, we implement this term using the sub-gradient [17]. We trained SCE using Scaled Conjugate Gradient Descent [64] on the full training set. Like any neural network-based model, the hyperparameters of SCE need to be tuned for optimum performance. Table 2 contains the list with the range of values we used in this research. We used validation set to choose the optimal

Table 2 Hyperparameters for sparsity-promoted centroid-encoder

Hyper parameter	Range of values
#SCG iteration	{25, 50, 75, 100}
#Hidden layers (L)	{1, 2}
#Hidden nodes (H)	{50, 100, 200, 250, 500}
Activation function	Hyperbolic tangent (tanh)
λ	{0.01, 0.001, 0.0001, 0.0002, 0.0004, 0.0006, 0.0008}
#Center/Class	{1, 2, 3, 4, 5}

value. For a small sample size data set (high-dimensional biological data), we ran a five-fold cross validation on the training set to pick the optimum value. The Python implementation of Sparsity-promoted Centroid-Encoder is available in Github repository https://github.com/Tomojit1/SparseCentroid-Encoder.git.

3.2.1 Feature cut-off

The 1-norm of the sparse layer (SPL) drives a lot of weight to near zero. Often hard thresholding or a ratio of two consecutive weights is used to pick the nonzero weight [5]. We take a different approach. After training SCE, we create a *sparsity curve* from the weights of the sparse layer by arranging the absolute value of the weights in descending order, then find the elbow of the curve. We measure the distance of each point on the curve to the straight line formed by joining the first and last points of the curve. The point with the largest distance is the position (P) of the elbow. We pick all the features whose absolute weight is greater than that of P. We demonstrate the feature cut-off in Fig. 2 with high-dimensional SMK_CAN data, which has 19,993 genes per sample. The two panels display the absolute weight of the sparsity promoting layer in descending order for two runs. The red dot indicates the exact position of P determined by the cut-off algorithm.

3.3 Empirical analysis of SCE

In this section, we present an empirical analysis of our model. The results of feature selection for the digits 5 and 6 from the MNIST set are displayed in Fig. 3. In panel (a), we compare the two terms that contribute to Eq. 4, i.e., the centroid-encoder and ℓ_1 costs, weighted with different values of λ. As expected, we observe that the CE cost monotonically decreases with λ, while the ℓ_1 cost increases as λ decreases. For larger values of λ, the model focuses more on minimizing the ℓ_1-norm of the sparse layer, which results in smaller values. In contrast, the model pays more attention to minimizing the CE cost for small λs; hence, we notice smaller CE cost and higher ℓ_1 cost.

Panel (b) of Fig. 3 shows the accuracy on a validation set as a function nine different values of λ; the validation accuracy reached its peak for $\lambda = 0.001$. In panels (c) and (d), we plotted the magnitude of the feature weights of the sparse layer in descending order. The sharp decrease in the

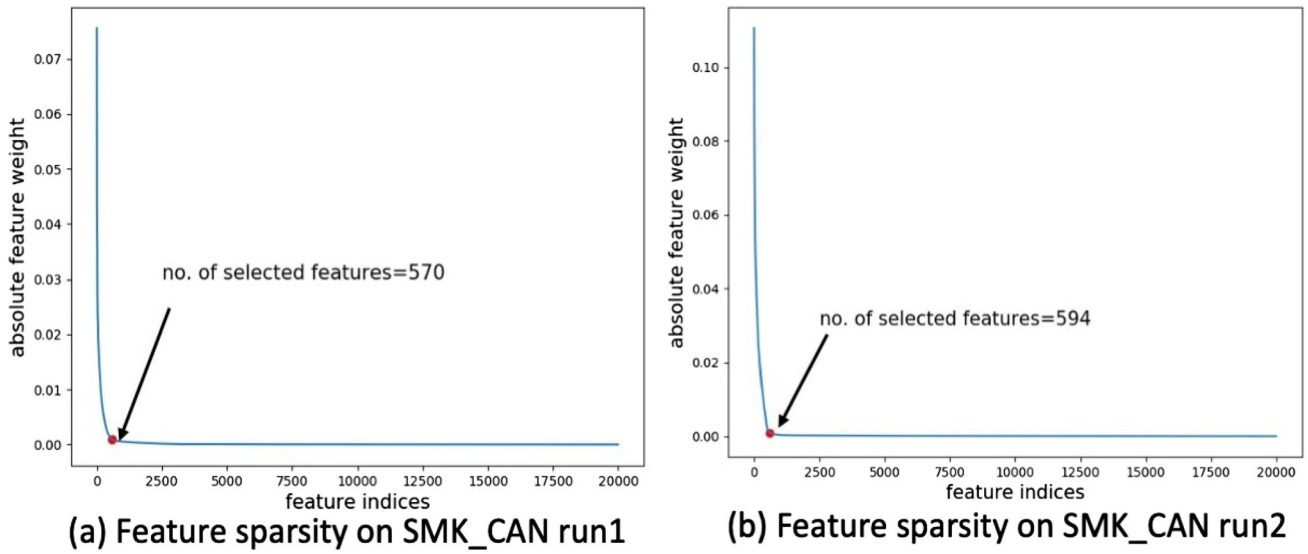

Fig. 2 Figure to display the absolute weight SPL layer in descending order over two run on high dimensional SMK_CAN data. The experiment is done with $\lambda = 0.0002$

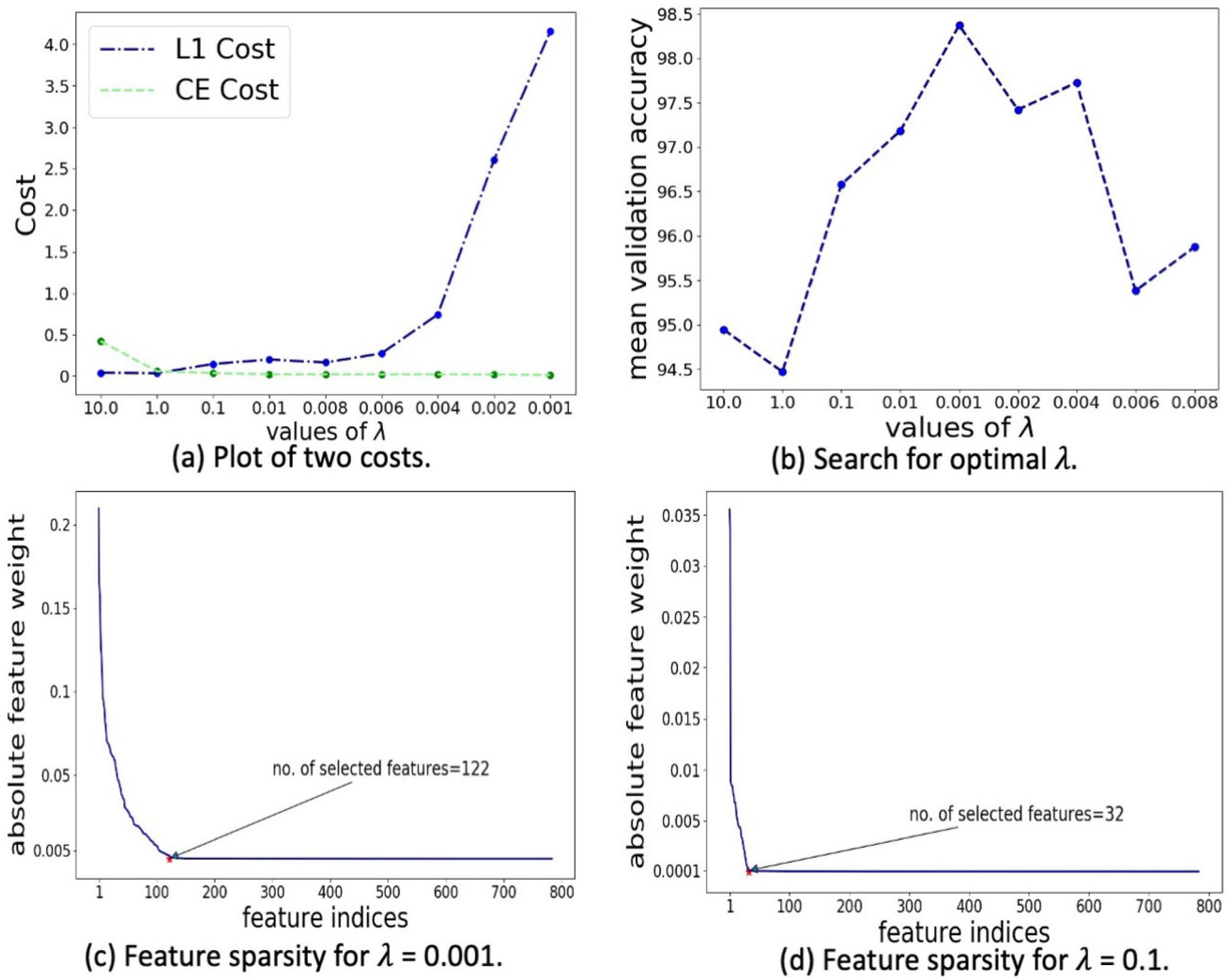

Fig. 3 Analysis of SCE. **a** Change of the two costs over λ. **b** Change of validation accuracy over λ. **c** Sparsity plot of the weight of W_{SPL} for $\lambda = 0.001$. **d** Same as **c** but $\lambda = 0.1$

magnitude of the weights demonstrates the promotion of sparsity by SCE. The model effectively ignores features by setting their weight to approximately zero. Notice the model produced a sparser solution for $\lambda = 0.1$, selecting only 32 features compared to 122 chosen variables for $\lambda = 0.001$. Figure 4 shows the position of the selected features, i.e., pixels, on the digits 5 and 6. The intensity of the color represents the feature's importance. Dark blue signifies a higher absolute value of the weight, whereas light blue means a smaller absolute weight.

Our next analysis shows how SCE extracts informative features from a multi-modal data set, i.e., data sets whose classes appear to have multiple clusters. In this case, one center per class may not be optimal, e.g., ISOLET data. To this end, we trained SCE using a different number of centers per class where the centers were determined using standard k-Means algorithm [65, 66]. After the feature selection, we calculated the validation accuracy and plotted it against the number of centers per class in Fig. 5. The validation accuracy jumped significantly from one center to two centers per class. The increased accuracy indicates that the speech classes are multi-modal, further validated by the two-dimensional PCA plot of the three classes shown in panel (b)–(d).

4 Feature selection workflow using SCE

By design, sparse methods identify a small number of features that accomplish a classification task. If one is interested in *all* the discriminatory features that can be used to separate multiple classes, then, one can repeat the process of removing good features. This section describes how SCE can be used iteratively to extract all discriminatory features from a data set; see [5] for an application of this approach to sparse support vector machines.

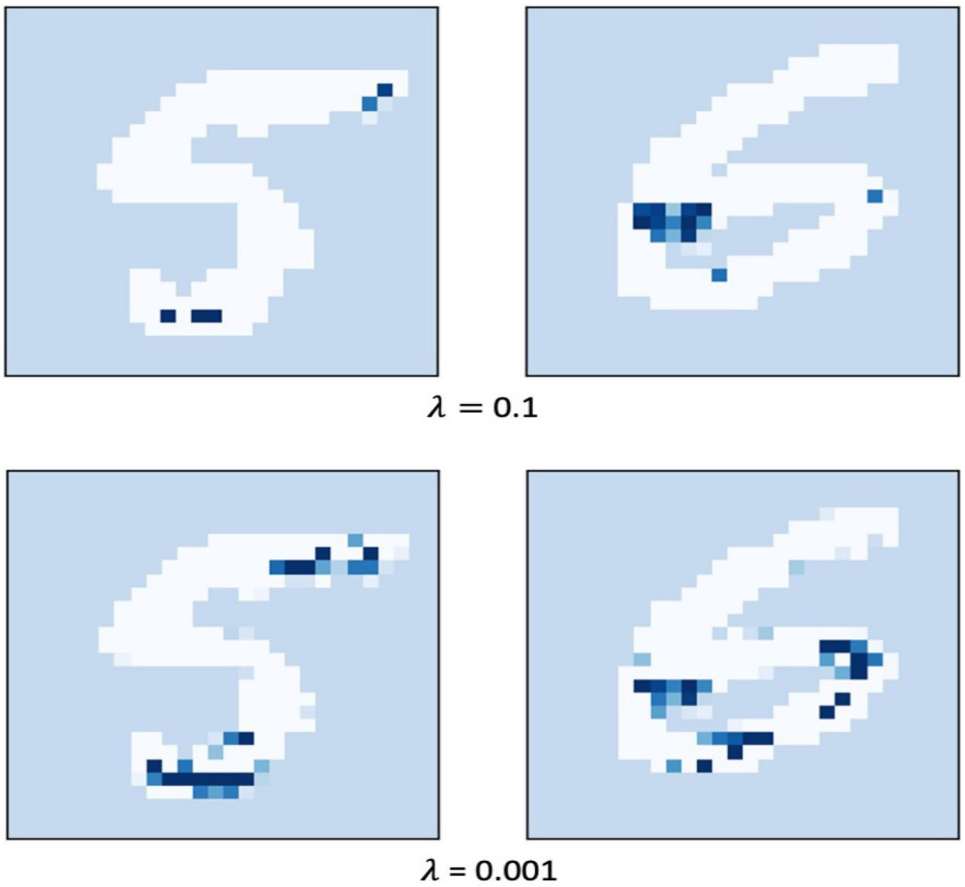

Fig. 4 Demonstration of the sparsity of the proposed model on MNIST digits 5 and 6. The digits are shown in white, and the selected pixels are marked using blue-the darkness of blue indicates the relative importance of the pixel to distinguish the two digits. We showed the selected pixels for two choices of λ. Notice that for $\lambda = 0.1$, the model chose the lesser number of features, whereas it picked more pixels for $\lambda = 0.001$. λ is the nob which controls the sparsity of the model

SCE is a model based on neural network architecture; hence, it is a non-convex optimization. As a result, multiple runs will produce different solutions, i.e., different feature sets on the same training set. These features may not be optimal given an unseen test set. To find out the robust features from a training set, we resort to frequency-based feature pruning. In this strategy, first, we divide the entire training set into k folds. On each of these folds, we ran the SCE and picked the top N (user select) number of features. We repeat the process T times to get $k \times T$ feature sets. Then, we count the number of occurrences of each feature and call this number the frequency of a feature. We ordered the features based on the frequency and picked the optimum number from a validation set. We present the feature selection workflow in Fig. 6.

5 Minimizing 1-norm using stochastic optimization

In this section, we investigate the challenges of minimizing ℓ_1-norm using stochastic optimization, such as stochastic gradient descent [SGD; 67], adaptive moment estimation [Adam; 68]. These techniques are beneficial for large-scale machine learning, see [69]. The authors of Group Sparse ANN [26] and DFS [25] used stochastic optimization with ℓ_1-norm on neural network architecture to promote feature sparsity. In recent work, Yamada et al. [37] reported that DFS and Group Sparse ANN failed to induce sparsity on several bench-marking feature selection data sets. However, the authors did not investigate the root cause. Note that stochastic optimizations like SGD, Adam require hyper-parameters, e.g., learning rate, mini-batch size, momentum. Calculating the gradient on a random subsample (mini-batches) of a training set might add noise that may affect ℓ_1-norm minimization. We did an array of experiments to evaluate the dependencies of the hyper-parameters on ℓ_1-norm minimization.

All the experiments in this section use Sparsity-promoted Centroid-Encoder on MNIST data set. This time we use Adam to optimize the network parameters over mini-batches. We used one hidden layer with 500 hyperbolic tangent ('tanh') activation units with a learning rate and λ set to 0.01 and 0.0001, respectively. In Fig. 7, we present the result of the first experiment, where we show the effect of the size of the mini-batch. The three columns (A, B, and C) show results for a specific choice of mini-batch, i.e., 512, 1024, and 5000. For each column, the upper panel

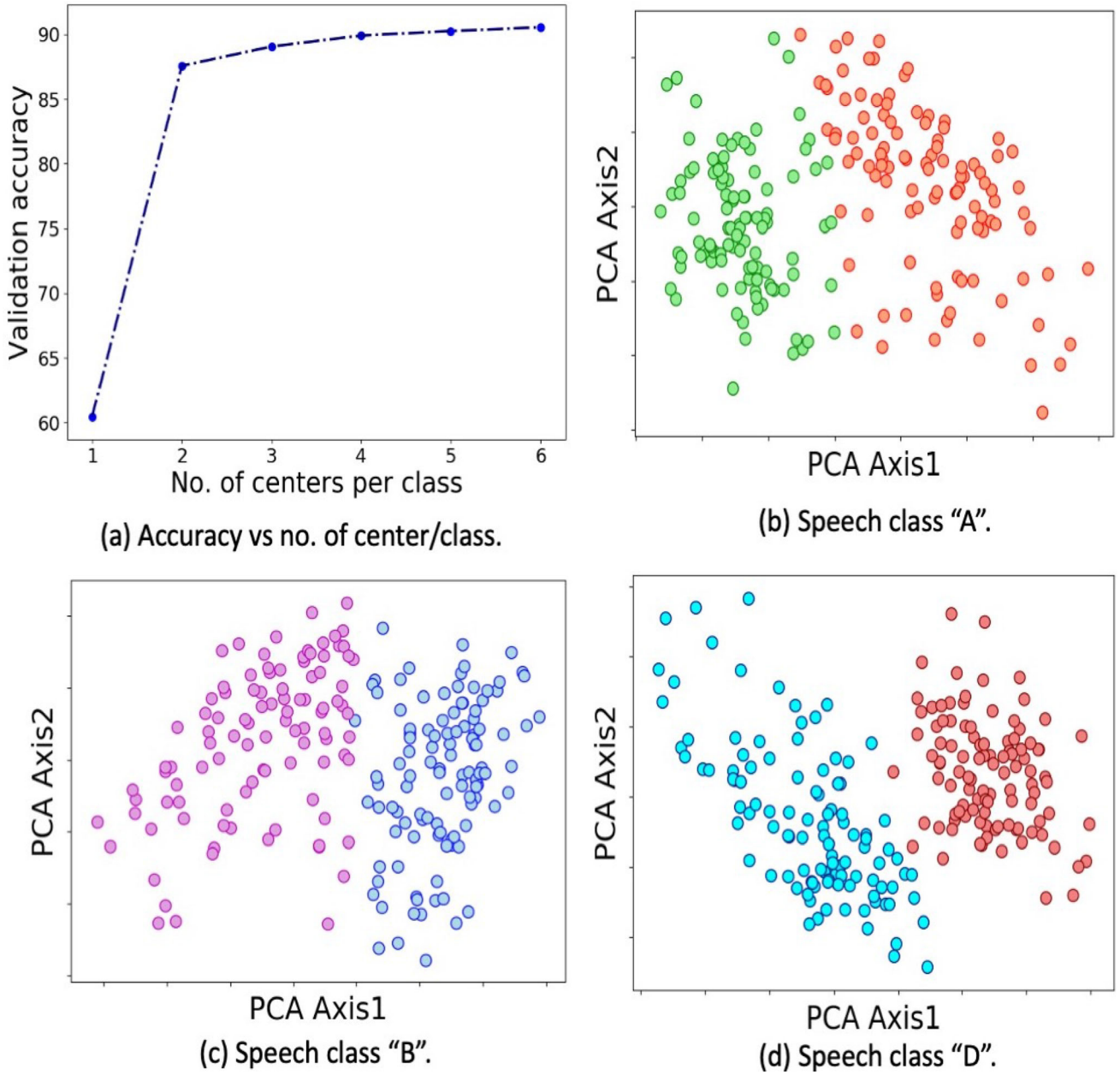

Fig. 5 Sparsity-promoted Centroid-Encoder for multi-modal data set. Panel **a** shows the increase in validation accuracy over the number of centroids per class. Panel **b–d** shows the two-dimensional PCA plot of the three speech classes

shows the position of the top 200 selected pixel, and the lower panel shows the absolute weight of the sparse layer in descending order.

Minimizing the ℓ_1-norm with smaller mini-batches (512) does not induce sparsity. Surprisingly, the ℓ_1-norm of the sparse layer put higher weight on the pixels around the border, ignoring the pixels in the center of the image. The model selects only 8 pixels (colored in teal) for mini bath 1024; among them, four pixels reside at the image's border. The position of the pixels and the sparsity plot improves significantly for mini-batch size 5000, selecting around 300 pixels from the center of the picture. Also, notice that the scale of the absolute weight increases with the size of the min-batch. We saw similar observations while working with a ReLU activation function, i.e., the relation between the mini-batch size and the sparsity does not change if we switch from tanh to ReLU.

Next, we show the effect of the learning rate/step size on the model's sparsity for $\lambda = 0.0001$ and mini-batch size of 5000. Figure 8 shows the results. For a relatively larger step size (0.1), the model did not induce sparsity, and a lot of selected pixels (top 200) lie on the border of the image,

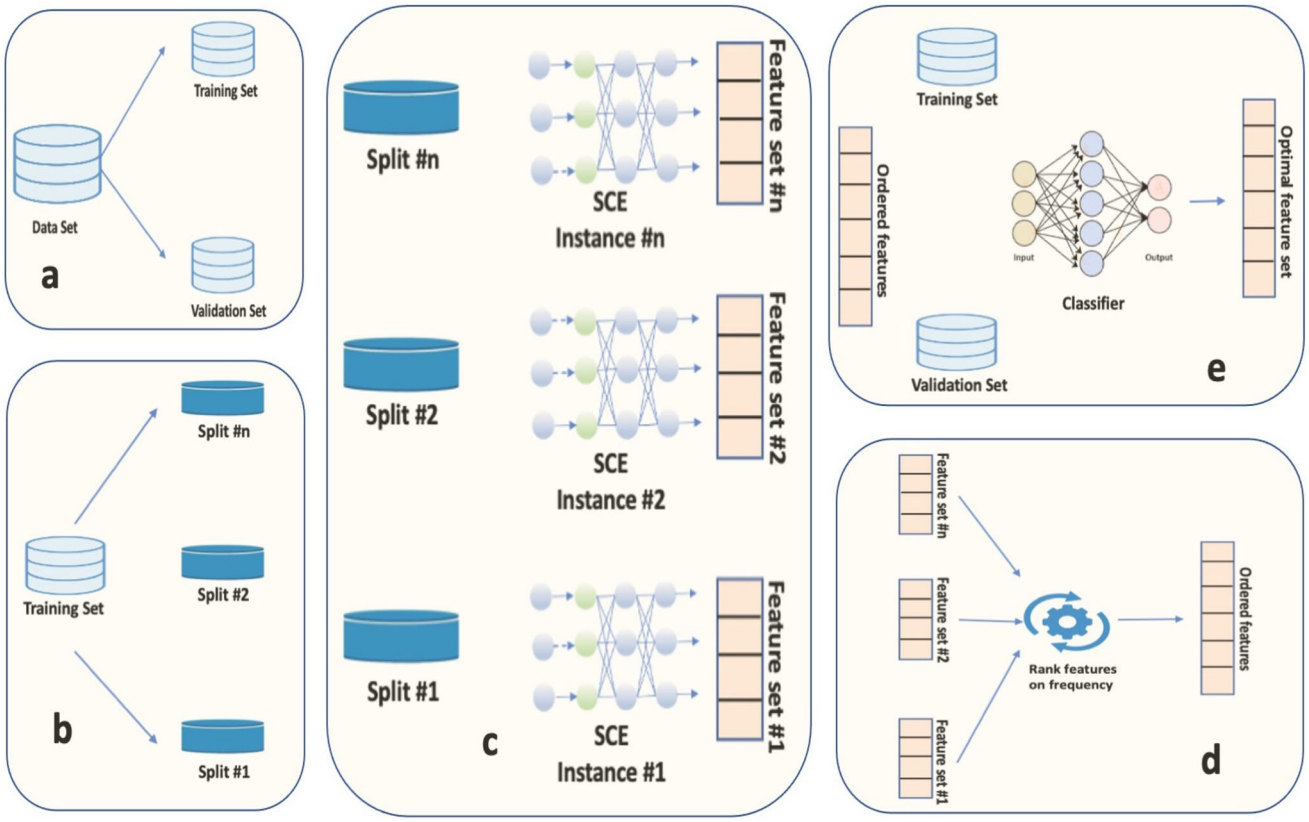

Fig. 6 Feature selection workflow using Sparsity-promoted Centroid-encoder: **a** First, the data set has been partitioned into training and validation. **b** We further partitioned the training set into n splits. **c** On each of the training splits, we ran Sparsity-promoted Centroid-encoder to get n feature sets. **d** We calculated the occurrence of each feature among the n sets and called it the frequency of the feature. We ranked features from high to a low frequency to get an ordered set. **e** At last, we picked the optimum number of features using a validation set

suggesting the presence of noise. In contrast, the model produces sparse solutions for learning rates of 0.01 and 0.001. In both cases, the selected pixels also reside in the middle of the image.

Figure 9 shows the result of the second experiment where we study the effect of penalty term λ for three different values 0.01 (panel A), 0.001 (panel B), and 0.0001 (panel C) for a fixed mini-batch size of 5000 and learning rate of 0.01. Notice that the model did not promote sparsity for $\lambda = 0.01, 0.001$. The ℓ_1-norm of the sparse layer selects pixels from all over the image when $\lambda = 0.01$; in contrast, $\lambda = 0.001$ ignores the middle of the images and picks pixels from the boundary. Interestingly, the selected pixels form a circle. Clearly, these two values of λ would not pick the most informative features from an MNIST image. On the other hand, we see a sparser solution for $\lambda = 0.0001$, selecting around 325 features from the middle of the 28×28 grid. The position of the selected pixels also makes sense as the digits lie in the center of the grid.

The in-depth analysis of this Section reveals an essential aspect of stochastic optimization when minimizing the ℓ_1-norm. We have observed that the hyper-parameters play a crucial role. Smaller mini-batches and higher λs do not promote feature sparsity, and the selected features perhaps contain noise. The learning rate also dictates the sparsity when other hyper-parameters are kept constant. These challenges can be overcome by carefully tuning the hyper-parameters using a validation set. So, in summary, minimizing ℓ_1-norm using stochastic optimization is challenging and requires a careful selection of hyper-parameters to induce feature sparsity. Consequently, we didn't use stochastic optimization while training SCE; instead, we used Scaled Conjugate Gradient (SCG) descent [64]. Unlike Adam or SGD, SCG calculates the step size/learning rate at each iteration, thus reducing the effort of tuning one hyperparameter. We also used the entire training set to calculate the gradient, which reduces the step of adjusting

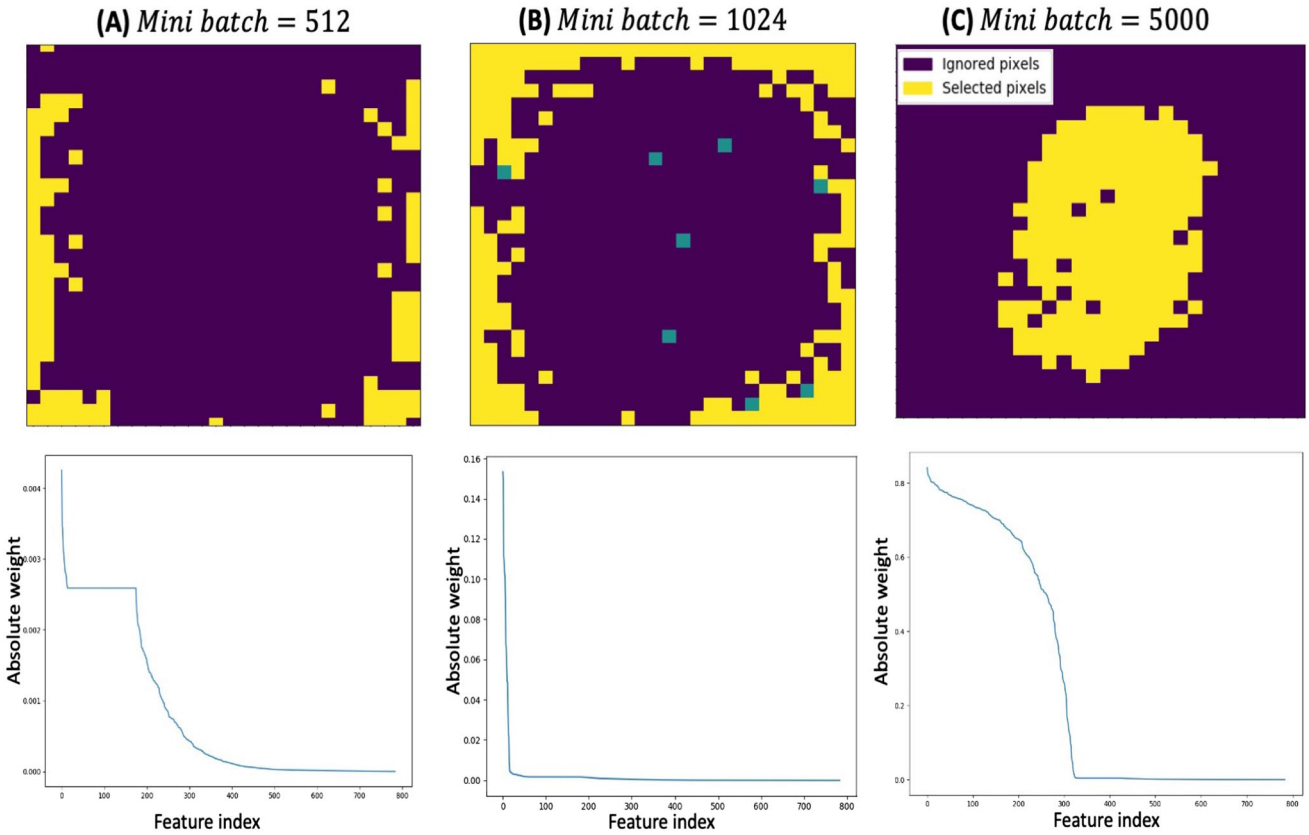

Fig. 7 Effect of the size of mini-batch on ℓ_1-norm minimization using SCE for three choices of mini-batches- 512 in (**A**), 1024 in (**B**), and 5000 in (**C**). For each case, the upper panel shows the position of the selected pixels in a 28 × 28 grid, and the lower panel presents the absolute weight of the sparse layer in descending order

the mini-batch size apart from the fact that the model parameters were updated on the actual gradient.

6 Experimental results

We present the comparative evaluation of our model on various data sets using several feature selection techniques.

6.1 Experimental details

We used twelve data sets from a variety of domains (image, biology, speech, and sensor; see Table 3) and five neural network-based models to run three benchmarking experiments. To this end, we picked the published results from four papers [25, 34, 36, 37] for benchmarking. We followed the same experimental methodology described in those papers for an apples-to-apples comparison. This approach permitted a direct comparison of LassoNet, FsNet, Supervised CAE, DFS, and Stochastic Gates using the authors' best results. All three experiments follow the standard workflow.

- Split each data sets into training and test partition.
- Run SCE on the training set to extract top $K \in \{10, 16, 50\}$ features.
- Using the top K features train a one hidden layer ANN classifier with H ReLU units to predict the test samples. The H is picked using a validation set.
- Repeat the classification 20 times and report average accuracy.

Now, we describe the details of the three experiments.

6.1.1 Experiment 1

The first bench-marking experiment is conducted on five real-world high dimensional biological data sets: ALLAML, GLIOMA, SMK_CAN, Prostate_GE, GLI_85, and CLL_SUB[1] to compare SCE with FsNet and Supervised CAE (SCAE). Following the experimental protocol of Singh et al. [36], we randomly partitioned each data into a 50:50 ratio of train and test and ran SCE on the training set. After that, we calculated the test accuracy using the top $K = \{10, 50\}$ SCE features. We repeated the experiment 20 times and reported the mean accuracy. We ran a 5-fold

[1] Available at https://jundongl.github.io/scikit-feature/.

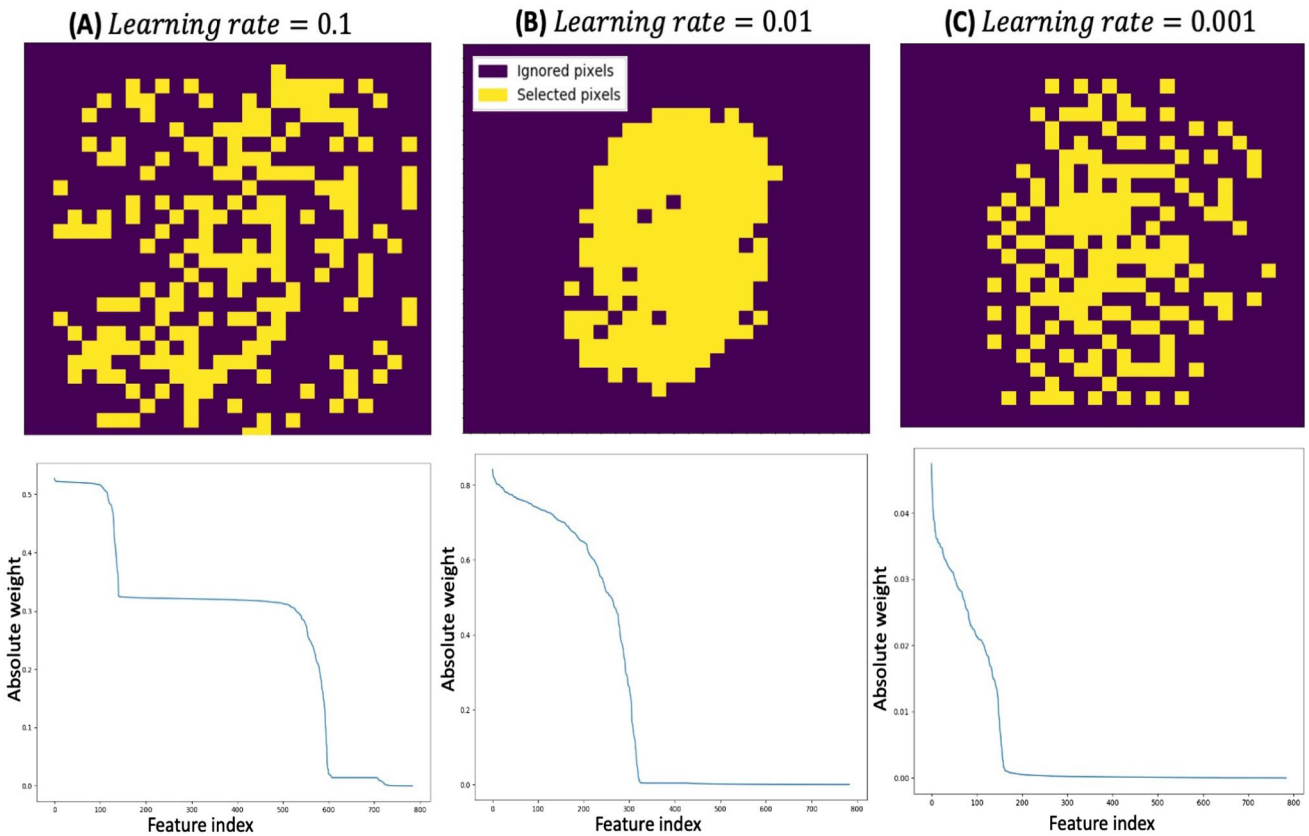

Fig. 8 Effect of learning rate on ℓ_1-norm minimization using SCE for three values 0.1 in (**A**), 0.01 in (**B**), and 0.001 in (**C**). For each case, the upper panel shows the position of the selected pixels in a 28×28 grid, and the lower panel presents the absolute weight of the sparse layer in descending order

cross-validation on the training set to tune the hyperparameters.

6.1.2 Experiment 2

In the second bench-marking experiment, we compared SCE with LassoNet [34] and Stochastic Gate [37] on six data sets: Mice Protein,[2] COIL20, Isolet, Human Activity, MNIST, and FMNIST.[3] Following the experimental set of Lemhadri et al. we split each data set into 70:10:20 ratio of training, validation, and test sets. We ran SCE on the training set to pick the top $K = 50$ features to predict the class labels of the sequester test set. We extensively used the validation set to tune the hyperparameters.

6.1.3 Experiment 3

In the last benchmark, we used the single cell GM12878 data[4] which has separate training, validation, and test sets. The SCE is run to select the top $K = 16$ features to compare the prediction performance with Deep/Shallow DFS [25], and Lasso. Again, we used the validation set for hyperparameters tuning.

6.2 Results

Now, we discuss the results of the three benchmarking experiments. In Table 4, we present the results of the first experiment where we compare SCE, SCAE, and FsNet on five high-dimensional biological data sets. Apart from the results using a subset (10 and 50) of features, we also provide the prediction using all the features. In most cases, feature selection helps improve classification performance. Generally, SCE features perform better than SCAE and FsNet; out of the twelve classification tasks, SCE produces the best result on eight. Notice that the top fifty SCE features give a better prediction rate than the top ten in all the cases. Interestingly, the accuracy of SCAE and FsNet drop significantly on SMK_CAN, GLI_85, and CLL_SUB using the top fifty features.

Now, we turn our attention to the results of the second experiment, as shown in Table 5. The features of the

[2] There are some missing entries that are imputed by mean feature values.

[3] Available at UCI Machine Learning repository.

[4] The authors of [25] shared the data with us.

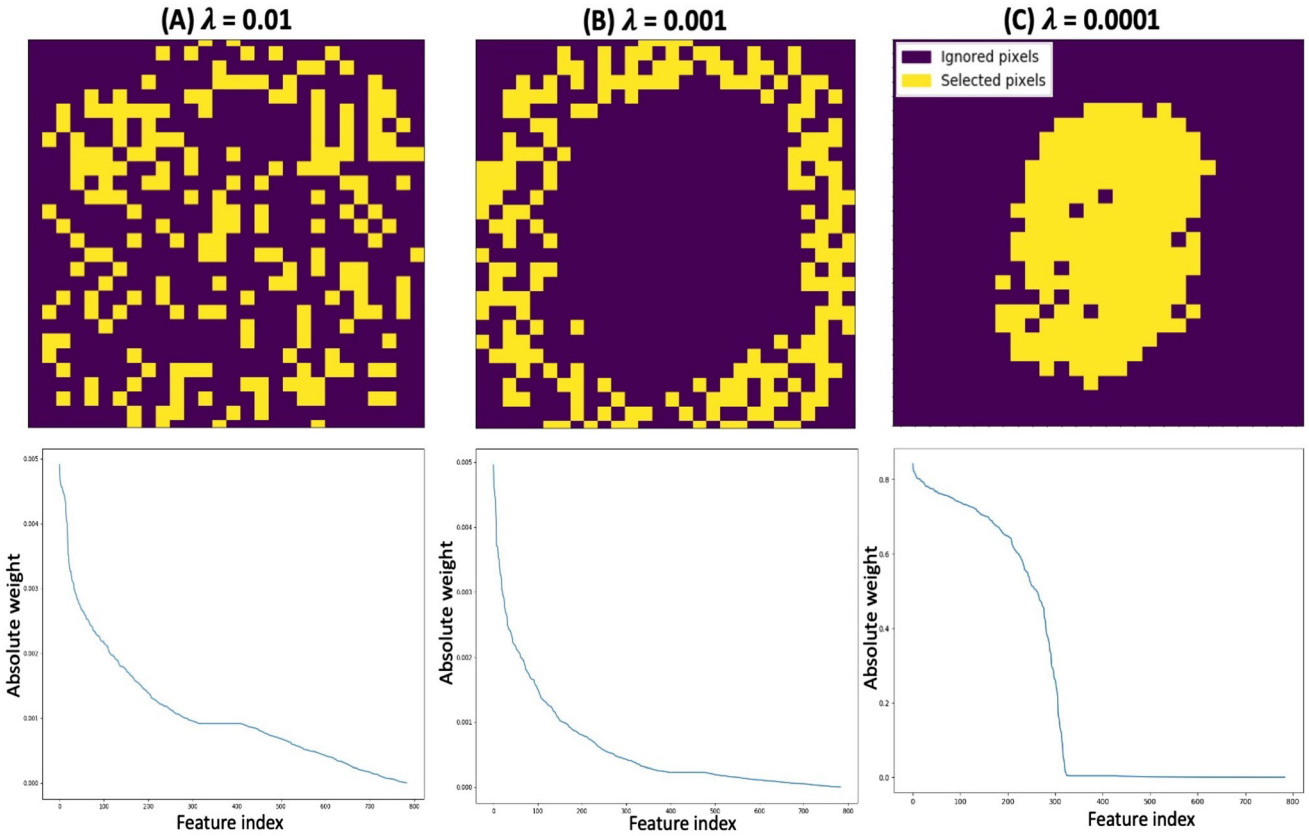

Fig. 9 Effect of λ on ℓ_1-norm minimization using SCE for three values 0.01 in (**A**), 0.001 in (**B**), and 0.0001 in (**C**). For each case, the upper panel shows the position of the selected pixels in a 28 × 28 grid, and the lower panel presents the absolute weight of the sparse layer in descending order

Table 3 Descriptions of the data sets used for benchmarking experiments

Data set	#Features	#Classes	#Samples	Domain
ALLAML	7129	2	72	Biology
GLIOMA	4434	4	50	Biology
SMK_CAN	19,993	2	187	Biology
Prostate_GE	5966	2	102	Biology
GLI_85	22,283	2	85	Biology
CLL_SUB	11,340	3	111	Biology
GM12878	93	3	6468	Biology
Mice protein	77	8	975	Biology
COIL20	1024	20	1440	Image
Isolet	617	26	7797	Speech
Human activity	561	6	5744	Accelerometer sensor
MNIST	784	10	70,000	Image
FMNIST	784	10	70,000	Image

The source of the data sets is mentioned in the specific experiment below

Sparsity-promoted Centroid-Encoder produce better classification accuracy than LassoNet and STG in all the cases. Especially for Mice Protein, Activity, Isolet, FMNIST, and MNIST, our model has better accuracy by 2.5–4.5%. The results for Stochastic Gates (STG) in [37] are not in a table form, but our eyeball comparison of classification accuracy with the top 50 features on ISOLET, COIL20, and MNIST suggests that stochastic gate is not more accurate than SCE. For example, using the top 50 features, STG obtains approximately 85% accuracy on ISOLET,

Table 4 Comparison of mean classification accuracy of FsNet, SCAE, and SCE features on five real-world high-dimensional biological data sets

Data set	Top 10 features			Top 50 features			All features
	FsNet	SCAE	SCE	FsNet	SCAE	SCE	ANN
ALLAML	91.1	83.3	**92.5**	92.2	93.6	**95.9**	89.9
Prostate_GE	87.1	83.5	**89.5**	87.8	88.4	**89.9**	75.9
GLIOMA	62.4	58.4	**63.2**	62.4	60.4	**69.0**	70.3
SMK_CAN	**69.5**	68.0	68.6	64.1	66.7	**69.4**	65.7
GLI_85	87.4	**88.4**	84.1	79.5	82.2	**85.5**	79.5
CLL_SUB	**64.0**	57.5	53.1	**58.2**	55.6	55.6	56.9

The prediction rates are averaged over twenty runs on the test set. Numbers for FsNet and SCAE are being reported from [36]. On these data sets, SCE is run using one centroid per class. The best results are highlighted in bold

Table 5 Classification results using LassoNet, STG, and SCE features on six publicly available data sets

Data set	Top 50 features			#Centers	All features
	LassoNet	STG	SCE	for SCE	ANN
Mice Protein	95.8	NA	**98.4**	1	100.00
MNIST	87.3	91.0	**93.8**	3	97.60
FMNIST	80.0	NA	**84.7**	3	90.16
ISOLET	88.5	85.0	**91.1**	5	96.96
COIL-20	99.1	97.0	**99.3**	1	98.87
Activity	84.9	NA	**89.4**	4	92.81

The column '#Centers for SCE' denotes how many centroids per class are used to train SCE. Numbers for LassoNet and STG are reported from [34] and [37], respectively. All the reported accuracies are measured on the test set. NA means the result has not been reported. We highlighted the best results in bold

while SCE obtains 91.1%; STG obtains about 97% on COIL20, while SCE obtains 99.3%; on the data set, MNIST STG achieves approximately 91%, while SCE 93.8%. In this experiment, we ran SCE with multiple centroids per class and observed an improved prediction rate than one center per class on Isolet, Activity, MNIST, and FMNIST. The observation suggests that the classes are multi-modal, providing a piece of valuable information. The optimum number of centers was picked using the validation set.

In Table 6, we present the results of our last experiment on the single cell data GM12878. We use the published results for deep feature selection (DFS), shallow feature selection, and Lasso from the work of Li et al. to evaluate SCE. To compare with Li et al. we used the top 16 features to report the mean accuracy of the test samples. We see that the SCE features outperform all the other models. Among all the models, Lasso exhibits the worst performance with an accuracy of 81.86%. This relatively low accuracy is not surprising, given Lasso is a linear model.

6.3 Experiment with feature selection workflow

This Section presents the results using the feature selection workflow mentioned in Sect. 4. To this end, we used a high-dimensional biological data set, GSE73072, that has proven to be very valuable for understanding the human immune response to respiratory infection. The framework is applied to SCE and Random Forest (RF), a widely used feature selection tool in Computational Biology. The details of the data set and the experimental results are given below.

GSE73072 is a microarray data set which is a collection of gene expressions taken from human blood samples as part of multiple clinical challenge studies [70] where individuals were infected with the following respiratory viruses HRV, RSV, H1N1, and H3N2. In our experiment, we excluded the RSV study. Blood samples were taken from the individuals before and after the inoculation. RMA normalization [71] is applied to the entire data set, and the LIMMA [72] is used to remove the subject-specific batch effect. Each sample is represented by 22,277 probes associated with gene expression. The data are publically available on the NCBI GeneExpression Omnibus (GEO) with identifier GSE73072.

We conducted an experiment on this data where the goal is to predict the classes control, shedders, and non-shedders at the very early phase of the infection, i.e., at time bin spanning hours 1–8. Controls are the pre-infection samples, whereas shedders and non-shedders are post-infection samples picked from the time bin 1–8 hr. Shedders actually disseminate virus, while non-shedders do not. We considered six studies, including two H1N1 (DEE3, DEE4), two H3N2 (DEE2, DEE5), and two HRV (Duke, UVA) studies. We used 10% training samples as a validation set-the training set comprised all the studies except for the DEE5, which was kept out for testing. We did a leave-one-subject-out (LOSO) cross-validation on the test set using the selected features from the training set. The validation set is

Table 6 Classification accuracies using the top 16 features by various techniques

Data set	Top 16 features			
	SCE	Deep DFS	Shallow DFS	Lasso
GM12878	**87.51**	85.67	85.34	81.86

Results of Deep DFS, Shallow DFS, Lasso, and Random Forest are reported from [25]. The best classification result is highlighted in bold

used to find the optimum number of features and to tune model-specific hyper-parameters. As our primary goal is to determine the utility of the proposed feature selection framework, we compared the results with two sets of features for each model. The first set is computed with the framework, and the send set is derived without the framework.

The results of this data set are shown in Table 7. Notice that features computed with the framework generalize the test samples better for both SCE and RF. The framework improved the classification of the DEE5 study by a margin of 17 and 10% for SCE and RF, respectively. Not only that, the variance of the balanced accuracy decreases for both models. It is pretty fascinating that with the framework, the models picked a relatively small number of features, 35 for SCE and 30 for RF, out of 22,277 genes to achieve relatively high accuracy. Note that the optimal number of features is picked using the validation set. Finally, we point out that SCE features with the framework have a performance benefit over RF.

6.4 Analysis of results

The experimental results in Sects. 6.2 and 6.3 show that Sparsity-promoted Centroid-Encoder features often perform better than other state-of-the-art methods on diverse sets. Here, we analyze the features in more detail to explain the improved performance of our model. We visualize the selected features directly or indirectly to explain and interpret their discriminative quality. To start with, we show the position of the selected pixels from the MNIST images in Fig. 10 over two runs using the whole training set.

Table 7 Balanced success rate (BSR) of LOSO cross-validation on the DEE5 test set. The selected features from training set are used to predict the classes of control, shedder, and non-shedder. The best classification result is highlighted in bold

Time bin	Model	No. of features	BSR
1–8	SCE with workflow	35	**88.59 ± 2.12**
	SCE without workflow	35	71.15 ± 5.39
	RF with workflow	30	82.65 ± 2.51
	RF without workflow	30	72.78 ± 6.36

The model picks almost the same number of pixels (194 and 198) across two runs, with 167 overlapping ones. Most of the selected pixels reside in the middle of the image, making sense as the MNIST digits lie in the center of a 28 × 28 grid. Notice that the non-overlapping pixels of the two runs are neighbors, making sense as the neighboring pixels perhaps contain similar information about the digits.

Now, we turn our attention to a high-dimensional biological data set, ALLAML, which has 7129 genes per sample. The top 10 and 50 SCE features predicted the test samples with an accuracy of more than 92%, better than the accuracy of using all the genes. Unlike MNIST, we cannot visualize the selected features directly on a grid, so we use an indirect method to interpret the discriminative power of the features. Here, we use PCA, a widely used unsupervised linear projection technique, to visualize the data using all features vs. the SCE features. Figure 11 presents the three-dimensional visualization of ALLAML data using all genes and the top 50 SCE genes. The projection with all 7129 genes in panel (a) does not separate the classes entirely. On the other hand, the projection using the top 50 SCE genes does separate the two classes better. Notice that the test cases are mapped close to the corresponding training class, which explains why SCE features to produce high test accuracy. These examples not only demonstrate the discriminative quality of the SCE features but also help us to interpret the features.

The classification performance gives a quantitative measure that does not reveal the biological significance of the selected genes. We did a literature survey of the top genes selected by sparsity-promoted centroid-encoder on GM12878 single cell data and provided a detailed description in the appendix. Some of these genes play an essential role in transcriptional activation, e.g., H4K20ME1 [73], TAF1 [74], H3K27ME3 [75]. Gene H3K27AC [76] plays a vital role in separating active enhances from inactive ones. Besides that, many of these genes are related to the proliferation of the lymphoblastoid cancer cells, e.g., POL2 [77], NRSF/REST [78], GCN5 [79], PML [80]. This survey indicates the possible biological significance of the selected genes.

7 Discussion and conclusion

In this paper, we proposed a novel feature selection technique Sparsity-promoted Centroid-Encoder. Using the basic multi-layer perceptron neural network architecture, the model backpropagates the Centroid-Encoder cost to a feature selection layer which filters out non-discriminating

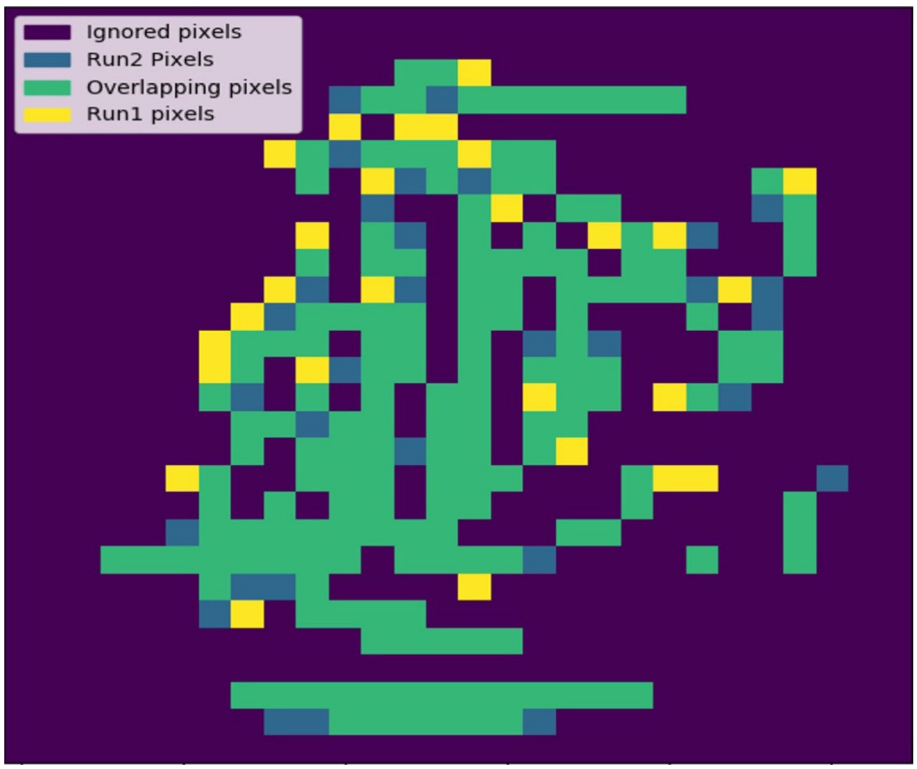

Fig. 10 Pixels selected by Sparsity-promoted Centroid-Encoder on MNIST with all the ten classes. The figure shows the position of the selected pixels over two runs ($\lambda = 0.0002$) on a 28×28 grid. SCE ignores the boundary of the image and picks most of the pixels from the middle, making sense as the MNIST digits lie in the center of a 28×28 grid. Notice that there is a significant number of common pixels across the two runs and the non-overlapping pixels reside in neighboring space

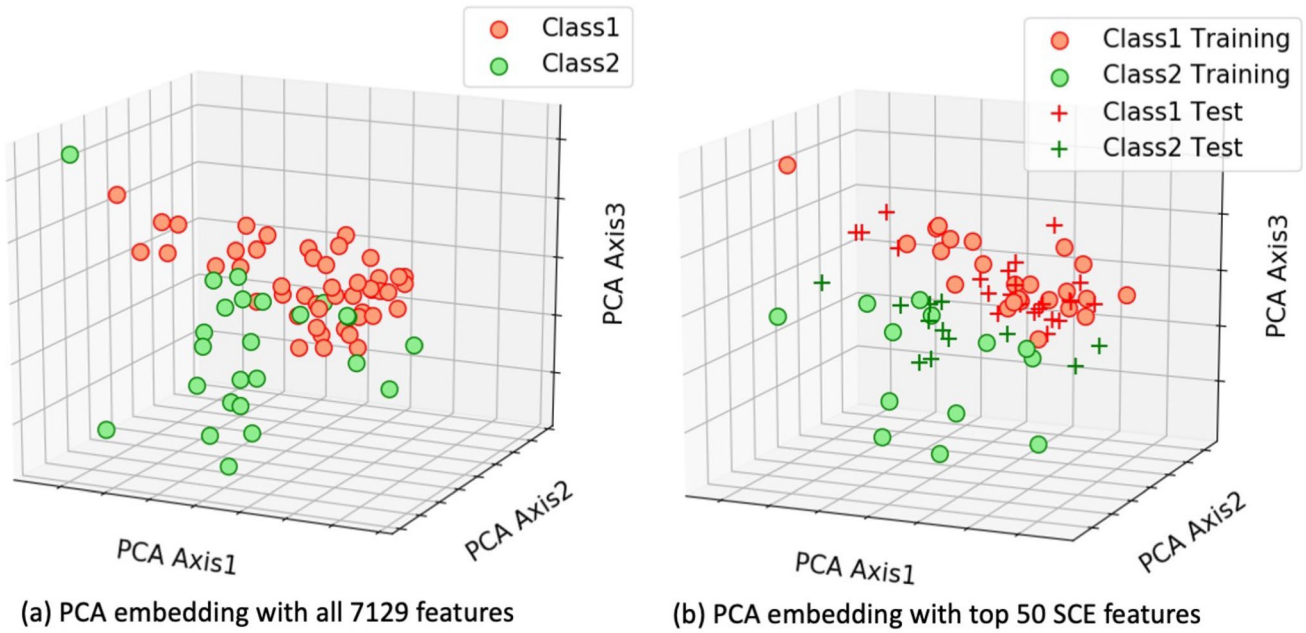

Fig. 11 Three-dimensional embedding using PCA on ALLAML data. **a** The entire data set is projected on the first three principal components. **b** First, the data set is partitioned into training and test sets by a 50:50 ratio. After that, the training and test samples are restricted to the top 50 SCE features, which are computed from the training set. The training set is used to calculate the principal components, and then, the training and the test samples are projected on the first three principal components

features by ℓ_1-regularization. The setting allows the feature selection to be data-driven without the need for any prior knowledge, such as the number of features to be selected and the underlying distribution of the input features. One key attribute of our method is the ability to model intra-class variance with multiple centroids per class. This approach improves the discriminative power of the selected

features and offers new information about the data set, i.e., whether the classes are unimodal or multi-modal.

The in-depth empirical analysis of the SCE provides the interpretability of our model. For example, the interplay between the CE cost and the ℓ_1 cost of SPL explains how the model should behave over different choices of the weighting parameter λ. A higher value of λ forces the model to minimize the 1-norm allowing the CE cost to increase. On the other hand, a lower λ decreases the CE cost allowing the 1-norm to grow. As an effect, the model selects fewer features for relatively large values of the parameter λ, and vice versa, demonstrated by the MNIST digits. We chose the optimal λ from a wide range of values using the validation set and observed that smaller values work better for our model. We noted that the variability in the loss function varied smoothly with λ making selection of an optimal parameter more robust. The ℓ_1 penalty on the SPL layer induces sharp sparsity on the high-dimensional SMK_CAN data without shrinking all the variables. Our feature cut-off technique correctly demarcates the crucial features from the rest.

We established the utility of our feature selection model using several benchmarking experiments involving seven methods. The results span fourteen data sets from various domains, such as image, speech, accelerometer sensors, and biology, providing evidence that the features of SCE produce better generalization performance than previously state-of-the-art models. We compared SCE with FsNet, primarily designed for high-dimensional biological data and found that our proposed method outperformed it in most cases. SCE also compares favorably with a supervised version of the Concrete Autoencoder (SCAE). Note that FsNet and SCAE use differentiable techniques, i.e, employ smooth cost functions. In data sets with more samples than the number of variables, SCE produces better classification results than LassoNet, Stochastic Gate, and DFS.

SCE may employ multiple centroids to capture the variability within a class, improving the prediction rate of unknown test samples. In particular, the prediction rate on the ISOLET improved significantly from one centroid to multiple centroids, suggesting that the speech classes are multi-modal. The two-dimensional PCA of ISOLET classes further confirms the multi-modality of the classes. We also observed an enhanced classification rate on MNIST, FMNIST and Activity data with multiple centroids. In contrast, using a single-center per class performed better for other data sets (e.g., COIL-20, Mice Protein, GM12878). Hence, apart from producing an improved prediction rate, our model can provide extra information about whether the data is unimodal or multi-modal. This aspect of sparsity-promoted centroid-encoder distinguishes it from the STG, CAE LassoNet, and DFS, which do not model the multi-modal nature of the data.

We demonstrated the interpretability of the selected features using several examples. The visualization of the MNIST pixels and the PCA projection of ALLAML data provided a qualitative explanation for a high prediction rate. On MNIST digits, SCE selected most of the pixels from the central part of the image, ignoring the border, making sense as the digits lie in the center of a 28×28 grid. On the high-dimensional ALLAML data, the top 50 SCE features showed better class separability on three-dimensional PCA space. Besides the visual explanation, the survey of the sixteen SCE genes of GM12878 data indicates plausible biological significance.

We also presented a feature selection workflow to determine the optimal number of robust features. Our experimental results showed that the prediction rate using the workflow improves the generalization performance for Random Forest and SCE. We think the workflow will benefit other nonconvex methods. We have presented a detailed empirical analysis to point out the challenges of inducing feature sparsity using stochastic optimization. Although the study is done using Sparsity-promoted Centroid-Encoder, it is plausible that other neural network-based models may exhibit similar behavior, an area of research we hope to explore in the future.

The Sparsity-promoted Centroid-Encoder presented here produced state-of-the-art results on many benchmarking data sets. Nonetheless, we think the performance of the model can potentially be improved further by exploring additional extensions. Currently, the model maps a sample to its class centroid while applying sparsity. The features may not be discriminatory if two class centroids are close in the ambient space. Adopting a cost that also caters to separating the classes may be beneficial. Our model may be extended to a semi-supervised feature selection regime by combining the centroid-encoder cost with the autoencoder loss. In the future, we will explore these ideas.

Appendix 1 Data set specific SCE configuration

In the following Table, we present the SCE architecture used for each data sets.

See Table 8.

Table 8 Details of network topology and hyperparameters for SCE

Data set	Network topology	Activation	λ	SCG iteration
ALLAML	$d \rightarrow 100 \rightarrow d$	tanh	0.0002	50
GLIOMA	$d \rightarrow 100 \rightarrow d$	tanh	0.0006	60
SMK_CAN	$d \rightarrow 500 \rightarrow d$	tanh	0.0002	50
Prostate_GE	$d \rightarrow 100 \rightarrow d$	tanh	0.0008	50
GLI_85	$d \rightarrow 250 \rightarrow d$	tanh	0.0008	75
Mice protein	$d \rightarrow 100 \rightarrow d$	tanh	0.001	25
GM12878	$d \rightarrow 50 \rightarrow 50 \rightarrow d$	tanh	0.01	50
COIL20	$d \rightarrow 100 \rightarrow d$	tanh	0.001	50
Isolet	$d \rightarrow 100 \rightarrow d$	tanh	0.001	50
Human activity	$d \rightarrow 100 \rightarrow d$	tanh	0.001	50
MNIST	$d \rightarrow 250 \rightarrow d$	tanh	0.0002	50
FMNIST	$d \rightarrow 250 \rightarrow d$	tanh	0.0004	50

The number d is the input dimension of the network and is data set dependent

Appendix 2 Biological significance of the selected genes of GM12878

- *POL2 (POLR2A)*: It is a subunit of RNA polymerase II, which interacts with nuclear CD26 using a chromatin immunoprecipitation assay. This interaction led to transcriptional repression of the POLR2A gene, resulting in a proliferation of cancer cells [77].
- *H4K20ME1* This gene has been implicated in transcriptional activation. Recent studies showed a strong correlation between H4K20me1 and gene activation in the regions downstream of the transcription start site [73].
- *NRSF (REST)* NRSF/REST is highly expressed in non-neuronal tissues like the lung. The findings of Kreisler et at. [78] support that NRSF/REST may act as an essential modulator of malignant progression in small-cell lung cancer.
- *TAF1* It is the largest integral subunit of TFIID, initiates RNA polymerase II-mediated transcription. Wang et al. discovered a critical promoter-binding function of TAF1 in transcription regulation [74].
- *H3K27AC* This gene distinguishes active enhancers from inactive/poised enhancer elements containing H3K4me1 alone [76].
- *GCN5* GCN5 functions as a transcriptional coactivator of E2f1 target genes. In small-cell lung cancer, E2F1 recruits GCN5 to acetylate H3K9, facilitating transcription of E2F1, CYCLIN E, and CYCLIN D1 (39) all of which promote cellular proliferation and tumor growth [79].
- *PML* The PML gene provides instructions for a protein that acts as a tumor suppressor, which means it prevents cells from growing and dividing too rapidly or in an uncontrolled way [80].
- *RUNX3* This gene binds to the core DNA sequence 5′-PYGPYGGT-3′ found in several enhancers and promoters. It also interacts with other transcription factors. It functions as a tumor suppressor, and the gene is frequently deleted or transcriptionally silenced in cancer [83].
- *ZZZ3* It's protein binding gene which often promotes gene activation [84].
- *H3K27ME3* This gene can function as silencers to regulate gene expression [75].

Acknowledgements We would like to acknowledge support for this research from the National Science Foundation under award NSF-ATD 1830676.

Data availability The data sets used in Experiment 1 can be found at https://jundongl.github.io/scikit-feature/. See the reference at [81]. Data related to Experiment 2 can be found at UCI Machine Learning Repository [82]. The Experiment 3 was run on the data available at https://github.com/yifeng-li/DECRES.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interest to disclose.

References

1. Pease AC, Solas D, Sullivan EJ, Cronin MT, Holmes CP, Fodor SP (1994) Light-generated oligonucleotide arrays for rapid DNA sequence analysis. Proc Natl Acad Sci 91(11):5022–5026. https://doi.org/10.1073/pnas.91.11.5022
2. Shalon D, Smith SJ, Brown PO (1996) A DNA microarray system for analyzing complex DNA samples using two-color fluorescent probe hybridization. Genome Res 6(7):639–645
3. Metzker ML (2010) Sequencing technologies-the next generation. Nat Rev Genet 11(1):31
4. Reuter JA, Spacek DV, Snyder MP (2015) High-throughput sequencing technologies. Mol Cell 58(4):586–597
5. O'Hara S, Wang K, Slayden RA, Schenkel AR, Huber G, O'Hern CS, Shattuck MD, Kirby M (2013) Iterative feature removal yields highly discriminative pathways. BMC Genomics 14(1):832
6. Lazar C, Taminau J, Meganck S, Steenhoff D, Coletta A, Molter C, Schaetzen V, Duque R, Bersini H, Nowe A (2012) A survey on filter techniques for feature selection in gene expression microarray analysis. IEEE ACM Trans Comput Biol Bioinform 9(4):1106–1119
7. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
8. Yu L, Liu H (2003) Feature selection for high-dimensional data: a fast correlation-based filter solution. In: Proceedings of the 20th international conference on machine learning (ICML-03), pp 856–863
9. Vergara JR, Estévez PA (2014) A review of feature selection methods based on mutual information. Neural Comput Appl 24(1):175–186
10. Fleuret F (2004) Fast binary feature selection with conditional mutual information. J Mach Learn Res 5(9):1531–1551
11. El Aboudi N, Benhlima L (2016) Review on wrapper feature selection approaches. In: 2016 international conference on engineering & MIS (ICEMIS), pp 1–5. IEEE
12. Hsu C-N, Huang H-J, Dietrich S (2002) The annigma-wrapper approach to fast feature selection for neural nets. IEEE Trans Syst Man Cybern Part B (Cybernetics) 32(2):207–212
13. Goldberg David E, Henry H (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
14. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4, pp 1942–1948. IEEE
15. Lal TN, Chapelle O, Weston J, Elisseeff A (2006) Embedded methods. Feature extraction. Springer, Berlin, pp 137–165
16. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc Ser B Stat Methodol 58(1):267–288
17. Boyd S, Xiao L, Mutapcic A (2003) Subgradient methods. In: Lecture notes of EE392o, Stanford University, Autumn Quarter 2004, 2004–2005
18. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res 12(7):2121–2159
19. Candes EJ, Romberg JK, Tao T (2006) Stable signal recovery from incomplete and inaccurate measurements. Commun Pure Appl Math J Issued Courant Inst Math Sci 59(8):1207–1223
20. Candes EJ, Tao T (2005) Decoding by linear programming. IEEE Trans Inf Theory 51(12):4203–4215
21. Fonti V, Belitser E (2017) Feature selection using lasso. VU Amsterdam Research Paper in Business Analytics, vol 30, pp 1–25
22. Muthukrishnan R, Rohini R (2016) Lasso: a feature selection technique in predictive modeling for machine learning. In: 2016 IEEE International conference on advances in computer applications (ICACA), pp 18–20. IEEE
23. Kim Y, Kim J (2004) Gradient lasso for feature selection. In: Proceedings of the twenty-first international conference on machine learning, p 60
24. Chepushtanova S, Gittins C, Kirby M (2014) Band selection in hyperspectral imagery using sparse support vector machines. In: Velez-Reyes M, Kruse FA (eds) Algorithms and technologies for multispectral, hyperspectral, and ultraspectral imagery XX. Proc. of SPIE, vol 9088
25. Li Y, Chen C-Y, Wasserman WW (2016) Deep feature selection: theory and application to identify enhancers and promoters. J Comput Biol 23(5):322–336
26. Scardapane S, Comminiello D, Hussain A, Uncini A (2017) Group sparse regularization for deep neural networks. Neurocomputing 241:81–89
27. Li G, Gu Y, Ding J (2022) ℓ_1 regularization in two-layer neural networks. IEEE Signal Process Lett 29:135–139. https://doi.org/10.1109/LSP.2021.3129698
28. Zou H (2006) The adaptive lasso and its oracle properties. J Am Stat Assoc 101(476):1418–1429
29. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc Ser B Stat Methodol 67(2):301–320
30. Tibshirani RJ (2013) The lasso problem and uniqueness. Electron J Stat 7:1456–1490
31. Ghosh T, Ma X, Kirby M (2018) New tools for the visualization of biological pathways. Methods 132:26–33. https://doi.org/10.1016/j.ymeth.2017.09.006
32. Ghosh T, Kirby M (2022) Supervised dimensionality reduction and visualization using centroid-encoder. J Mach Learn Res 23(20):1–34
33. Bergstra J, Breuleux O, Bastien F, Lamblin P, Pascanu R, Desjardins G, Turian J, Warde-Farley D, Bengio Y (2010) Theano: a CPU and GPU math expression compiler. In: Proceedings of the python for scientific computing conference (SciPy), vol 4, pp.1–7
34. Lemhadri I, Ruan F, Abraham L, Tibshirani R (2021) Lassonet: a neural network with feature sparsity. J Mach Learn Res 22(127):1–29
35. Balın MF, Abid A, Zou J (2019) Concrete autoencoders: differentiable feature selection and reconstruction. In: International conference on machine learning, pp 444–453. PMLR
36. Singh D, Climente-González H, Petrovich M, Kawakami E, Yamada M (2020) Fsnet: feature selection network on high-dimensional biological data. arXiv preprint arXiv:2001.08322
37. Yamada Y, Lindenbaum O, Negahban S, Kluger Y (2020) Feature selection using stochastic gates. In: International conference on machine learning, pp 10648–10659. PMLR
38. Hoerl AE, Kennard RW (1970) Ridge regression: biased estimation for nonorthogonal problems. Technometrics 12(1):55–67
39. Marafino BJ, Boscardin WJ, Dudley RA (2015) Efficient and sparse feature selection for biomedical text classification via the elastic net: application to ICU risk stratification from nursing notes. J Biomed Inform 54:114–120
40. Shen L, Kim S, Qi Y, Inlow M, Swaminathan S, Nho K, Wan J, Risacher SL, Shaw LM, Trojanowski JQ (2011) Identifying neuroimaging and proteomic biomarkers for MCI and AD via the elastic net. In: International workshop on multimodal brain image analysis. Springer, pp 27–34
41. Sokolov A, Carlin DE, Paull EO, Baertsch R, Stuart JM (2016) Pathway-based genomics prediction using generalized elastic net. PLoS Comput Biol 12(3):1004790
42. Lindenbaum O, Steinerberger S (2021) Randomly aggregated least squares for support recovery. Signal Process 180:107858
43. Candes EJ, Wakin MB, Boyd SP (2008) Enhancing sparsity by reweighted l1 minimization. J Fourier Anal Appl 14(5):877–905
44. Daubechies I, DeVore R, Fornasier M, Gunturk CS (2010) Iteratively reweighted least squares minimization for sparse

recovery. Commun Pure Appl Math J Issued Courant Inst Math Sci 63(1):1–38
45. Bertsimas D, Copenhaver MS, Mazumder R (2017) The trimmed lasso: sparsity and robustness. arXiv preprint arXiv:1708.04527
46. Xie H, Huang J (2009) Scad-penalized regression in high-dimensional partially linear models. Ann Stat 37(2):673–696
47. Cortes C, Vapnik V (1995) Support vector machine. Mach Learn 20(3):273–297
48. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46(1):389–422
49. Mangasarian OL (1999) Arbitrary-norm separating plane. Oper Res Lett 24(1–2):15–23
50. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
51. Shaban WM (2022) Insight into breast cancer detection: new hybrid feature selection method. Neural Comput Appl 1–23
52. Yang X-S, Hossein Gandomi A (2012) Bat algorithm: a novel approach for global engineering optimization. Eng Comput 29(5):464–483
53. Dai L, Zhang J, Du G, Li C, Wei R, Li S (2023) Toward embedding-based multi-label feature selection with label and feature collaboration. Neural Comput Appl 35(6):4643–4665
54. Vahmiyan M, Kheirabadi M, Akbari E (2022) Feature selection methods in microarray gene expression data: a systematic mapping study. Neural Comput Appl 34(22):19675–19702
55. Meier L, Van De Geer S, Bühlmann P (2008) The group lasso for logistic regression. J R Stat Soc Ser B Stat Methodol 70(1):53–71
56. Kim SG, Theera-Ampornpunt N, Fang C-H, Harwani M, Grama A, Chaterji S (2016) Opening up the blackbox: an interpretable deep neural network-based classifier for cell-type specific enhancer predictions. BMC Syst Biol 10(2):243–258
57. Roy D, Murty KSR, Mohan CK (2015) Feature selection using deep neural networks. In: 2015 international joint conference on neural networks (IJCNN), pp 1–6. IEEE
58. Han K, Wang Y, Zhang C, Li C, Xu C (2018) Autoencoder inspired unsupervised feature selection. In: 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 2941–2945. IEEE
59. Taherkhani A, Cosma G, McGinnity TM (2018) Deep-fs: a feature selection algorithm for deep Boltzmann machines. Neurocomputing 322:22–37
60. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
61. Romero A, Carrier PL, Erraqabi A, Sylvain T, Auvolat A, Dejoie E, Legault M, Dubé M, Hussin JG, Bengio Y(2017) Diet networks: thin parameters for fat genomics. In: 5th international conference on learning representations, ICLR 2017, Toulon, France, 24–26 Apr, 2017, conference track proceedings. OpenReview.net. https://openreview.net/forum?id=Sk-oDY9ge
62. Al-Obeidat F, Tubaishat A, Shah B, Halim Z (2022) Gene encoder: a feature selection technique through unsupervised deep learning-based clustering for large gene expression data. Neural Comput Appl 1–23
63. Aminian M, Ghosh T, Peterson A, Rasmussen A, Stiverson S, Sharma K, Kirby M (2021) Early prognosis of respiratory virus shedding in humans. Sci Rep 11(1):1–15
64. Møller MF (1993) A scaled conjugate gradient algorithm for fast supervised learning. Neural Netw 6(4):525–533
65. Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28(2):129–137
66. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol 1, pp 281–297
67. Robbins H, Monro S (1951) A stochastic approximation method. Ann Math Stat 22:400–407
68. Kingma D, Ba JL (1951) 3rd international conference on learning representations, ICLR 2015-conference track proceedings. In: International conference on learning representations (ICLR) Adam: a method for stochastic optimization. Go to Reference in Article x
69. Bottou L (2010) Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp 177–186. Springer
70. Liu T-Y, Burke T, Park LP, Woods CW, Zaas AK, Ginsburg GS, Hero AO (2016) An individualized predictor of health and disease using paired reference and target samples. BMC Bioinform 17(1):1–15
71. Irizarry RA, Hobbs B, Collin F, Beazer-Barclay YD, Antonellis KJ, Scherf U, Speed TP (2003) Exploration, normalization, and summaries of high density oligonucleotide array probe level data. Biostatistics 4(2):249–264
72. Ritchie ME, Phipson B, Wu D, Hu Y, Law CW, Shi W, Smyth GK (2015) limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res 43(7):47–47
73. Barski A, Cuddapah S, Cui K, Roh T-Y, Schones DE, Wang Z, Wei G, Chepelev I, Zhao K (2007) High-resolution profiling of histone methylations in the human genome. Cell 129(4):823–837
74. Wang H, Curran EC, Hinds TR, Wang EH, Zheng N (2014) Crystal structure of a TAF1-TAF7 complex in human transcription factor IID reveals a promoter binding module. Cell Res 24(12):1433–1444
75. Cai Y, Zhang Y, Loh YP, Tng JQ, Lim MC, Cao Z, Raju A, Aiden EL, Li S, Manikandan L (2021) H3k27me3-rich genomic regions can function as silencers to repress gene expression via chromatin interactions. Nat Commun 12(1):1–22
76. Creyghton MP, Cheng AW, Welstead GG, Kooistra T, Carey BW, Steine EJ, Hanna J, Lodato MA, Frampton GM, Sharp PA (2010) Histone h3k27ac separates active from poised enhancers and predicts developmental state. Proc Natl Acad Sci 107(50):21931–21936
77. Mao C-G, Jiang S-S, Shen C, Long T, Jin H, Tan Q-Y, Deng B (2020) Bcar1 promotes proliferation and cell growth in lung adenocarcinoma via upregulation of polr2a. Thorac Cancer 11(11):3326–3336
78. Kreisler A, Strissel P, Strick R, Neumann S, Schumacher U, Becker C (2010) Regulation of the NRSF/REST gene by methylation and CREB affects the cellular phenotype of small-cell lung cancer. Oncogene 29(43):5828–5838
79. Yin Y-W, Jin H-J, Zhao W, Gao B, Fang J, Wei J, Zhang DD, Zhang J, Fang D (2015) he histone acetyltransferase GCN5 expression is elevated and regulated by c-Myc and E2F1 transcription factors in human colon cancer. Gene Expr 16(4):187
80. Salomoni P, Pandolfi PP (2002) The role of PML in tumor suppression. Cell 108(2):165–170
81. Li J, Cheng K, Wang S, Morstatter F, Trevino RP, Tang J, Liu H (2018) Feature selection: a data perspective. ACM Comput Surv (CSUR) 50(6):94
82. Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml
83. Yu G-P, Ji Y, Chen G-Q, Huang B, Shen K, Wu S, Shen Z-Y (2012) Application of RUNX3 gene promoter methylation in the diagnosis of non-small cell lung cancer. Oncol Lett 3(1):159–162
84. Mi W, Zhang Y, Lyu J, Wang X, Tong Q, Peng D, Xue Y, Tencer AH, Wen H, Li W (2018) The ZZ-type zinc finger of ZZZ3 modulates the ATAC complex-mediated histone acetylation and gene activation. Nat Commun 9(1):1–9