Journal of Machine Learning Research 23 (2022) 1-34 Submitted 2/20; Revised 12/21; Published 1/22

Supervised Dimensionality Reduction and Visualization using Centroid-Encoder

Tomojit Ghosh TOMOJIT.GHOSH@COLOSTATE.EDU
Department of Computer Science
Colorado State University
Fort Collins, CO 80523 ,USA

Michael Kirby KIRBY@MATH.COLOSTATE.EDU
Department of Mathematics
Colorado State University
Fort Collins, CO 80523, USA

Editor: Inderjit Dhillon

Abstract

We propose a new tool for visualizing complex, and potentially large and high-dimensional, data sets called Centroid-Encoder (CE). The architecture of the Centroid-Encoder is similar to the autoencoder neural network but it has a modified target, i.e., the class centroid in the ambient space. As such, CE incorporates label information and performs a supervised data visualization. The training of CE is done in the usual way with a training set whose parameters are tuned using a validation set. The evaluation of the resulting CE visualization is performed on a sequestered test set where the generalization of the model is assessed both visually and quantitatively. We present a detailed comparative analysis of the method using a wide variety of data sets and techniques, both supervised and unsupervised, including NCA, non-linear NCA, t-distributed NCA, t-distributed MCML, supervised UMAP, supervised PCA, Colored Maximum Variance Unfolding, supervised Isomap, Parametric Embedding, supervised Neighbor Retrieval Visualizer, and Multiple Relational Embedding. An analysis of variance using PCA demonstrates that a non-linear preprocessing by the CE transformation of the data captures more variance than PCA by dimension.

Keywords: Data Science, Machine Learning, supervised dimension reduction, centroid-encoder, autoencoder, variance.

1. Introduction

The ability *to see* in high-dimensions can assist data explorers in many ways, including the characterization of class structure, data separability, and experimental batch effects. One approach to data visualization is the process of mapping points in dimensions greater than three down to two or three dimensions, providing a window into high-dimensions so that they can be *seen* and interpreted. Principal component analysis (PCA; Pearson, 1901; Hotelling, 1933) continues to be one of the most widely used methods for data reduction and visualization over one hundred years after its initial discovery (Jolliffe and Cadima, 2016; Chepushtanova et al., 2020). Nonetheless, PCA often produces ambiguous results, in some cases collapsing distinct classes into overlapping regions in the setting where class labels are available. It is tempting to incorrectly infer from this that the data is not separable, even non-linearly, in higher dimensions.

Non-linear extensions to PCA were originally introduced to address the limitations of optimal linear mappings (Kramer, 1991b, 1992; Oja, 1991), also see (Kirby, 2001) for additional early references and details. These papers provided the first applications of autoencoder neural networks where data sets are non-linearly mapped to themselves. This is accomplished by first unfolding them in higher dimensions before passing through a bottleneck layer of a reduced dimension where data visualization is typically done. In the process of non-linear dimension reduction, novel *latent* or hidden features which are an amalgamation of the observables, may be discovered. A theoretical insight into these algorithms is provided by Whitney's theorem from differential geometry that connects autoencoders to manifold learning (Broomhead and Kirby, 2000, 2001). Fundamental innovations were proposed to these ideas in the setting of deep networks that made their application far more effective (Hinton and Salakhutdinov, 2006; Hinton et al., 2006).

Data reduction techniques such as PCA and its non-linear extensions have proven extremely useful for understanding data in high-dimensions, and in particular, as tools for visualization. In this paper we address the question of how such tools can be adapted to the case where class label information is available, i.e., supervised data reduction. While there has been considerable work on this topic related to linear methods, the focus in this paper is on non-linear labeled data reduction expressly for the purposes of data visualization. In this setting, the objective function of the optimization problem that encodes the goals of the dimension reduction also exploits information related to class membership.

More specifically, this paper concerns the integration of class label information to the non-linear autoencoder reduction process for the purposes of data visualization. Here we develop and comparatively evaluate this *centroid-encoder* (CE) algorithm designed for the analysis of labeled data. While standard autoencoders map the identity on points, centroid-encoders map points to their centroids while passing through a low-dimensional representation. This approach serves to provide a low-dimensional encoding for visualization while ensuring that elements with the same label retain their class structure. The smoothness of mapping functions ensures that a similar behavior is captured at the centroid-encoder bottleneck layer.

Here we summarize the main contributions of this paper. Centroid-encoder maps data to each class centroid through a low-dimensional bottleneck in a manner analogous to autoencoders. Hence it exploits label information in the non-linear data reduction process and may be viewed as a supervised variant of non-linear principal component analysis (Kramer, 1991b, 1992; Oja, 1991). As a mapping from the input space to the reduced space, CE generalizes directly to new data without the use of landmarks, or the computation of distance matrices typical of spectral methods (Belkin and Niyogi, 2003; Roweis and Saul, 2000) or self-organizing mappings (Kohonen, 1987; Ma et al., 2019). Given its bottleneck style architecture that implements a continuous non-linear mapping, CE is an effective tool for the visualization of class-labeled data where neighborhood relationships are transported from high to low-dimensions without explicitly computing the neighborhood graph in the ambient space. We show that CE produces similar or better results when compared with a range of other popular supervised and unsupervised visualization tools across a broad range of data sets. Our comparisons are both quantitative, using a classification score in the reduced space, as well as qualitative, with a visual comparison of neighborhood structures of well-known data sets. To provide insight into the effectiveness of the reduction model, we illustrate how it captures the variance of the data in the mapping layers. We apply CE to produce the first low-dimensional visualization of the large particle Physics Supersymmetry (SUSY) data set and show that it classifies samples with accuracy comparable to the best high-dimensional models. Lastly, we observe that CE allows for the use of multiple centroids per class and generalizes as a classifier to dimensions greater than three. However, our focus in this paper is on the application of CE for supervised data visualization.

The article is organized as follows: In Section 2, we review related work, for both supervised and unsupervised data visualization. In Section 3 we present the centroid-encoder algorithm and contrast it with auto-encoder data reduction. In Section 4 we apply CE to a range of bench-marking data sets taken from the literature. In Section 5, we analyze these results and conclude in Section 6.

2. Related Work

Data reduction for visualization has a long history and remains an area of active research. We describe the literature relating to unsupervised and supervised methods where we assume data class labels are unavailable and available, respectively.

This paper proposes a variation to the autoencoder, a dimension reducing mapping optimized to approximate the identity on a set of training data (Bourlard and Kamp, 1988; Kramer, 1991a; Hinton and Zemel, 1994; Kirby, 2001; Bengio et al., 2006). The autoencoder was initially proposed as a non-linear version of PCA (Kramer, 1991b, 1992). A comprehensive description of autoencoders and their variants can be found in (Goodfellow et al., 2016, chap. 14).

2.1 Unsupervised Methods

Principal Components Analysis is often the first (and last) tool used for visualization of unlabeled data and is designed to retain as much of the statistical variance as possible (PCA; Pearson, 1901; Hotelling, 1933), see also (Jolliffe, 1986). Equivalently, PCA minimizes the mean-square approximation error as well as Shannon's entropy (Watanabe, 1965). Self-organizing mappings (SOMs) learn non-linear *topology-preserving* transformations that map data points to centers that are then mapped to the center indices. SOMs have been widely used in data visualization, having been cited over 20,000 times since their introduction (Kohonen, 1982).

Another class of methods uses interpoint distances as the starting point for data reduction and visualization. For example, Multidimensional Scaling is a spectral method that computes a point configuration based on the computation of the eigenvectors of a doubly centered distance matrix (MDS; Torgerson, 1952). The goal of the optimization problem behind MDS is to determine a configuration of points whose Euclidean distance matrix is optimally close to the prescribed distance matrix. A related approach, known as Isomap, applies MDS to approximate geodesic distances computed numerically from data on a manifold (Isomap; Tenenbaum et al., 2000). Laplacian Eigenmaps is another popular spectral method that uses distance matrices to reduce dimension and conserve neighborhoods (LE; Belkin and Niyogi, 2003). These spectral methods belong to a class of techniques referred to as *manifold learning* algorithms; see also locally linear embedding (LLE; Roweis and Saul, 2000), stochastic neighbor embedding (SNE; Hinton and Roweis, 2003), and maximum variance unfolding (MVU; Weinberger and Saul, 2006).

The technique t-distributed stochastic neighbor embedding (t-SNE; van der Maaten and Hinton, 2008), an extension of SNE, was developed to overcome the data crowding or clumping problem often observed with manifold learning methods and is currently a popular method for data visualization. More recently, the uniform manifold approximation and projection (UMAP) algorithm has been proposed, which uses Riemannian geometry and fuzzy simplicial sets to create a low-dimensional and locally uniformly distributed embedding of the data (UMAP; McInnes et al., 2018). It has been reported that the algorithm has a better run time than t-SNE and offers compelling visualizations. Xie et al. developed a deeply embedded clustering (DEC) algorithm where the cluster assignment is optimized using an auxiliary target distribution on the output of the encoder (Xie et al., 2016). This unsupervised algorithm imposes a structure using Student's t-distribution to improve the clustering in the latent space. Note, the gradient update of the DEC algorithm requires distance calculation of a point to each of the cluster centers.

Note that these spectral methods including MDS, Laplacian Eigenmaps and UMAP compute embeddings based on solving an eigenvector problem requiring the entire data set. Such methods do not actually create mappings that can be applied to reduce the dimension of new data points without repeating the computation or resorting to the use of landmarks (De Silva and Tenenbaum, 2004). This, in contrast to methods such as PCA, SOM and autoencoders that serve as mappings for streaming data.

2.2 Supvervised Methods

Fisher's linear discriminant analysis (LDA) reduces the dimension of labeled data by simultaneously optimizing class separation and within-class scatter (see Fisher, 1936; Duda and Hart, 1973). Both LDA and PCA are linear methods in that they construct optimal projection matrices, i.e., linear transformations for reducing the dimension of the data.

It is, in general, possible to add labels to unsupervised methods to create their supervised analogues. A heuristic-based supervised PCA model first selects important features by calculating correlation with the labels and then applies standard PCA of the chosen feature set (Bair et al., 2006). Another supervised PCA technique, proposed by (Barshan et al., 2011), uses Hilbert-Schmidt independence criterion to compute the principal components which have maximum dependence on the labels. Colored maximum variance unfolding (MUHSIC; Song et al., 2007), which is the supervised version of MVU, is capable of separating different newsgroups better than MVU and PCA.

The projection of neighborhood component analysis (NCA; Goldberger et al., 2004) produces a better coherent structure in two-dimensional space than PCA on several UCI data sets. Dhillon et al. proposed *class preserving projection* of high dimensional data (Dhillon et al., 2002). Parametric embedding (PE; Iwata et al., 2007), which embeds high-dimensional data by preserving the class-posterior probabilities of objects, separates different categories of Japanese web pages better than MDS. Optimizing the NCA objective on a pre-trained deep architecture, Salakhutdinov et al. (nonlinear NCA; Salakhutdinov and Hinton, 2007) achieved 1% error rate on MNIST test data with 3-nearest neighbor classifier on 30-dimensional feature space. Min et al. proposed to optimize the NCA and maximally collapsing metric learning (MCML; Globerson and Roweis, 2006) objective using a Student t-distribution on a pre-trained network (t-Distributed Embedding; Min et al., 2010). Their approach yielded promising generalization error using a 5-nearest neighbor classifier on MNIST and USPS data on two-dimensional feature space. Neighbor retrieval visualizer (NeRV; Venna et al., 2010) optimizes its cost such that the similar objects are mapped close together in embedded space. Its supervised variant uses the class information to produce the low-dimensional embedding. NeRV and its supervised counterpart were reported to outperform some of the state-of-the-art dimension reduction techniques on a variety of datasets.

There are several proposed methods in literature where label information has been incorporated into the autoencoder architecture. For example, a supervised autoencoder (SAE) minimizes a bi-objective cost involving the unsupervised reconstruction loss and a supervised regression error on the sample labels (Le et al., 2018). Like SAE, Adversarial Autoencoder (AAE) (Makhzani et al., 2015) optimizes a dual objective in terms of the traditional reconstruction loss of autoencoders and adversarial criteria (Goodfellow et al., 2014) to match the aggregated posterior distribution of the autoencoder to an arbitrary prior distribution. The adversarial cost forces the encoder to learn the user-defined prior distribution; therefore, the decoder imposes the prior distribution while generating the samples. Labels are incorporated into the model to guide the data generation. Note AAE falls into the category of generative models. Nonlinear NCA (NNCA) optimizes the NCA cost (Goldberger et al., 2004) on a pre-trained (RBM) deep architecture (Salakhutdinov and Hinton, 2007) using pair-wise distances. The dt-NCA and dG-NCA algorithms also minimizes the NCA cost on a deep architecture, except a Gaussian, or Student's t-distribution is used to convert distances into probabilities (Min et al., 2010). Similarly dt-MCML and dG-MCML optimize the MCML cost (Globerson and Roweis, 2006) using a deep neural network where pair-wise distances are transformed into probabilities using Gaussian and Student's t-distribution correspondingly (Min et al., 2010).

Supervised Isomap (S-Isomap; Cheng et al., 2012), which explicitly uses the class information to impose dissimilarity while configuring the neighborhood graph on input data, has a better visualization and classification performance than Isomap. Zhang et al. suggested to use labels to optimize the objective of Local Linear Embedding and their supervised model (Zhang, 2009) performed better than LLE. Similarly, the supervised version of Laplacian Eigenmaps (Raducanu and Dornaika, 2012) yield better quality visualization than its unsupervised counterpart. Stuhlsatz et al. proposed a generalized discriminant analysis based on classical LDA (GerDA; Stuhlsatz et al., 2012), which is built on deep neural network architecture. Min et al. proposed a shallow supervised dimensionality reduction technique where the MCML objective is optimized based on some learned or precomputed exemplars (HOPE; Min et al., 2017). Although UMAP is originally presented as an unsupervised technique, its software package includes the option to build a supervised model [1].

Autoencoder neural networks are the starting point for our approach and are described in the next section. In a preliminary biological application, centroid-encoder (CE) was used to visualize the high-dimensional pathway data (Ghosh et al., 2018).

1. Manual located at https://readthedocs.org/projects/umap-learn/downloads/pdf/latest/.

2.3 Summary of Novelty

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

- CE uses the sample label to calculate the class-centroids in the ambient space and then maps the samples of a class to its centroid in the output layer.
- CE doesn't apply any cost function to the bottleneck layer.
- Models like SAE and AAE use multiple objectives to incorporate class labels. In contrast, CE uses a single objective to include supervision in training.
- SAE and AAE uses a reconstruction loss term that does not exploit label information. In contrast, CE employs label information in the reconstruction loss term.
- Unlike NNCA, dt/dG NCA, dt/dG MCML, and DEC Centroid-encoder doesn't require the calculation of pair-wise distances.
- Unlike AAE, the centroid-encoder doesn't employ any prior distribution in training.
- The DEC algorithm updates the cluster centers in the learning process, whereas the centroid-encoder works with a fixed set of centers, i.e., each class's centroid. DEC doesn't use class labels.

CE uses the sample label to calculate the fixed class-centroids in the ambient space and then maps the samples of a class through the encoder and decoder to their known centroids in the output layer. CE doesn't apply any cost to the bottleneck layer and uses a single objective based on reconstruction of the class means at the output. Hence, CE is a fully supervised training approach without any unsupervised, or semi-supervised characteristics. Unlike many approaches for visualization, CE captures global structures in the reduced space without the calculation of pair-wise distances or knowledge of prior distributions during training. The visualization of the high-dimensional data and their centroids is accomplished by the data reducing encoding.

3. The Centroid-Encoder

Here we propose a form of supervised autoencoder that exploits label information. The centroid-encoder (CE) is trained by mapping data through a neural network with the architecture of an autoencoder, but the learning target of any point is not that actual point, rather the mean of points in the associated class. So CE does not map the identity, now the target points in the image of the map are the centroids of the data in the ambient space. Centroid-encoder is a non-linear parametric map which minimizes the within-class variance by mapping all the samples of a class/category to its centroid.

Consider an M-class data set with classes denoted $C_j, j = 1, \ldots, M$ where the indices of the data associated with class C_j are denoted I_j. The centroid of each class is defined as

$$c_j = \frac{1}{|C_j|} \sum_{i \in I_j} x^i$$

$|C_j|$ is the cardinality of class C_j. The centroid-encoder is trained by determining a mapping f with the ordered input-output pairs

$$\{(x^i, c_j)\}_{i=1}^N$$

where c_j is the target center for data point x^i, i.e., $i \in I_j$.

The cost function of centroid-encoder is defined as

$$L_C(D;\theta) = \frac{1}{2N}\sum_{j=1}^{M}\sum_{i\in I_j}\|c_j - f(x^i;\theta))\|_2^2 \tag{1}$$

where $D = \{x^i\}$ is the data and θ represents all the unknown parameters, i.e., weights and biases, in the network. This cost is also referred to as the *distortion error* (Linde et al., 1980). In contrast, the cost function for the auto-encoder is

$$L_A(D;\theta) = \frac{1}{2N}\sum_{i=1}^{N}\|x^i - f(x^i;\theta))\|_2^2 \tag{2}$$

The parameters θ in each case are learned by using standard error backpropagation (Rumelhart et al., 1985; Werbos, 1974). The output errors Δ that are being back-propogated are

$$\Delta_C = \frac{1}{N}\sum_{j=1}^{M}(\sum_{i\in I_j} f(x^i;\theta) - c_j) \tag{3}$$

for centroid-encoder and

$$\Delta_A = \frac{1}{N}\sum_{i=1}^{N}(f(x^i,\theta) - x^i) \tag{4}$$

for auto-encoder. These terms Δ_A and Δ_C capture the difference between auto-encoder and centroid encoder; the gradient vector $\mathbf{g}$ is the same in both cases; see, e.g., (Goodfellow et al., 2016; Kirby, 2001) for details on the computation of the gradient $\mathbf{g}$.

The mapping f is modeled as the composition $f(\cdot) = h(g(\cdot))$ of a dimension reducing *encoder* mapping g followed by a dimension increasing *decoder* mapping h. The training goal is that a point gets mapped to its centroid for centroid-encoder

$$c_j \approx h(g(x^i))$$

or that a point gets mapped to itself for auto-encoder

$$x^i \approx h(g(x^i)).$$

The dimension reduced data then corresponds to the output of the encoder $y^i = g(x^i)$ for both CE and AE architectures. In this paper we are primarily concerned with visualization so the dimension of the range of g is 2, or 3; in principle there is no obstacle to taking values larger than this.

3.1 The Centroid-Encoder Training Algorithm

The training is very similar to that of an autoencoder and the steps are described in Algorithm 1. As with autoencoders, the centroid-encoder is a composition of two maps $f(x) = (h \circ g)(x)$. For visualization, we train a centroid-encoder network using a bottleneck architecture, meaning that the dimension of the image of the map g is 2 or 3.

Algorithm 1: Supervised Non-linear Centroid-Encoder (without pre-training).

```
  Input: Labeled data (D) = $\{x^i\}_{i=1}^N$ with M classes, $I_j$ index set of class $C_j$. User
         defined parameters: error tolerance $\tau$, learning rate $\mu$, bottleneck
         dimension $m$.
  Output: Bottleneck output $y^i = g(x^i)$; network parameters $\theta$.
  Result: Non-linear embedding of data in $m$ dimensions.
  Initialization: Class centroids $c_j = \frac{1}{|C_j|}\sum_{i \in I_j} x^i, j = 1, \ldots, M$. Iteration $t \leftarrow 0$.
                  Partition D into training (Tr) and validation set (V).
1 while $|L_C^{t+1}(V)$ - $L_C^t(V)| > \tau$ do
2     Compute the loss/error $L_C(V)$ and $L_C(Tr)$ using Equation 1
3     Compute backpropogation error $\Delta_C$ using Equation 3 on training set Tr.
4     Update model parameters $\theta$ by $\theta^{t+1} = \theta^t - \mu\Delta_C.\mathbf{g}^t$
5 end
```

3.1.1 Pre-training Centroid-encoder

Here we describe the pre-training strategy of centroid-encoder. We employ a technique we refer to as *pre-training with layer-freeze*, i.e., pre-training is done by adding new hidden layers while mapping samples of a class to its centroid; see Figure 1. In this approach, weights of hidden layers are learned sequentially. Initially $\theta = (W_1, \tilde{W}_1)$ and by the end of training $\theta = (W_1, \ldots, W_n, \tilde{W}_n, \ldots, \tilde{W}_1)$ using our θ notation for the set of unknown parameters. The algorithm starts by learning the parameters of the first hidden layer using standard error backpropagation (Rumelhart et al., 1985). Then the second hidden layer is added in the network with weights initialized randomly. At this point, the associated parameters of the first hidden layer are kept frozen. Now the parameters of the second hidden layer are updated using backpropagation. We repeat this step until pre-training is done for each hidden layer. Once pre-training is complete, we do an end-to-end fine-tuning by updating the parameters of all the layers at the same time. It's noteworthy that our pre-training approach is different than the greedy layer-wise pre-training proposed by (Hinton et al., 2006) and (Bengio et al., 2006, 2007). The unsupervised pre-training of Hinton et al. and Bengio et al. uses the activation of the l^{th} hidden layer as the input for the $(l+1)^{th}$ layer. In our approach, we use the original input-output pair (x^i, c_j) to pre-train each hidden layer. As we use the labels to calculate the centroids c_j, so our pre-training is supervised. In the future, we would like to explore the unsupervised pre-training in the setup of centroid encoding.

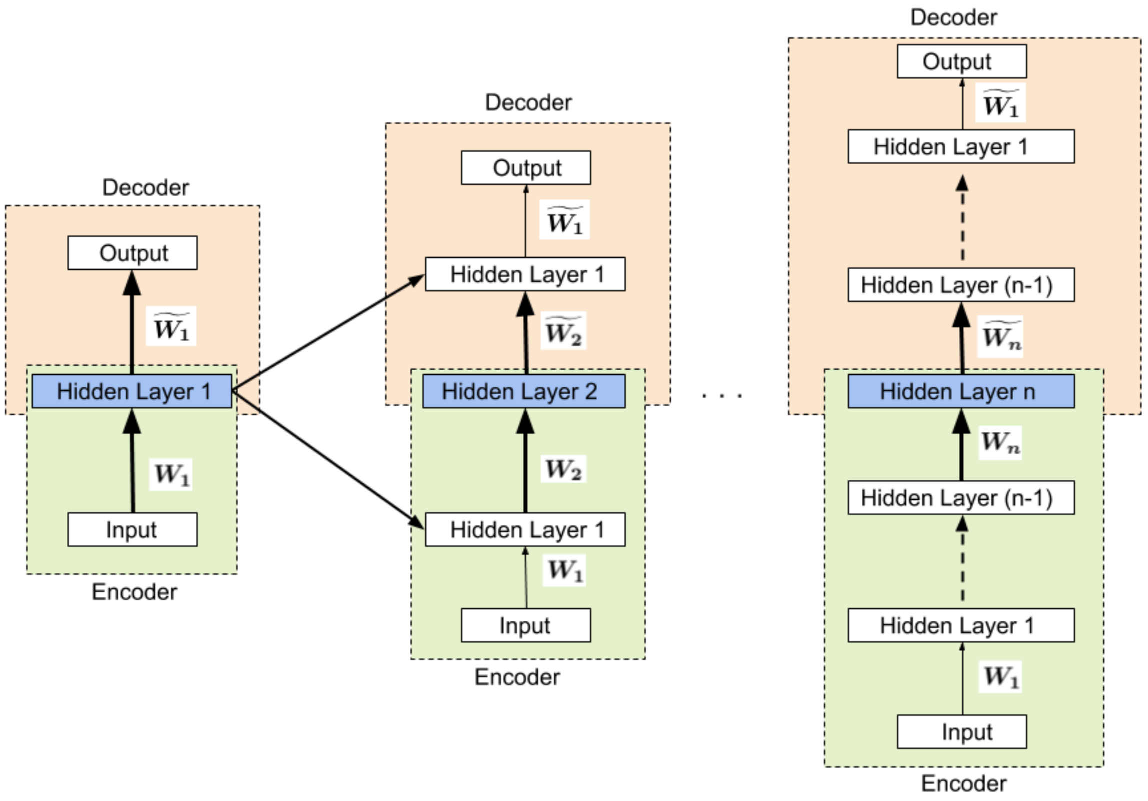


Figure 1: Pre-training a deep centroid-encoder by layer-freeze approach. In the first step, a centroid-encoder with the first hidden layer is pre-trained (left diagram). The pre-trained weights are $(W_1, \widetilde{W_1})$. In the next step, a new hidden layer is added by extending the network architecture. After that, the weights $(W_2, \widetilde{W_2})$ associated with the new hidden layer are updated while keeping the other weights $(W_1, \widetilde{W_1})$ fixed. This process is repeated to add more hidden layers.

4. Visualization Experiments

To evaluate centroid-encoder we select three suites of data sets from the literature and run three bench-marking experiments comparing centroid-encoder with a range of other supervised dimension reduction techniques. To objectively compare the performance of these supervised models we employ a standard class prediction error on the two-dimensional visualization domain.[2] The classification error is defined as

$$Error\ (\%) = \frac{100}{N}\sum_{i=1}^{N} I[l_i \neq f(\tilde{x_i})] \tag{5}$$

Here N is total number of test samples, l_i is the true label of the i^{th} test sample, and f is a classification function which returns the predicted label of the embedded test sample $\tilde{x_i}$. Here I denotes the indicator function. Following the supervised visualization literature

2. This corresponds to the bottleneck layer for CE.

(Goldberger et al., 2004; Venna et al., 2010; van der Maaten, 2009; Min et al., 2010), we chose the k-nearest neighbor (k-NN) as the classification function (f).

All results share the following workflow:

- Select a small data set from the training set and run 10-fold cross validation to determine the network architecture and hyper-parameters
- Using this architecture train K models on different data partitions as described below
- Using the sequestered test set compute average k-NN ($k = 5$) classification errors on the 2D representation with standard deviations

We follow this common evaluation strategy in the three bench-marking experiments so that our CE results are comparable to the published results. We describe details of each experiment including how the training and test partitions were selected below. In addition to prediction error, we also visualize the two-dimensional embedding of test samples of each model to do a subjective comparison. We fix the reduction dimension as 2 in all networks for our visualization application.

In Section 4.1, we provide the details of the three experiments. We show the implementation details in Section 4.2 and present the results in Section 4.3.

4.1 Experimental Details

The data sets we employed in our experiments are: (1) MNIST digits, (2) USPS data, (3) Letter Recognition data, (4) Landsat Satellite data, (5) Phoneme data, (6) Iris data, (7) Sonar data, and (8) Supersymmetry (SUSY) particle data. The details of the data sets may be found in the supplementary material. Here we describe the details of the three experiments.

Experiment 1: The first bench-marking experiment is conducted on the widely studied MNIST and USPS data sets. We compared CE with the following methods: autoencoder, non-linear NCA (Salakhutdinov and Hinton, 2007), supervised UMAP (McInnes et al., 2018), GerDA (Stuhlsatz et al., 2012), HOPE (Min et al., 2017), parametric t-SNE (van der Maaten, 2009), t-distributed NCA (Min et al., 2010), t-distributed MCML (Min et al., 2010), and supervised UMAP. All MNIST models were trained using the entire training set and evaluated on the standard test set. For USPS, we followed the strategy in (Min et al., 2010), where we randomly split the entire data set into a training set of 8000 samples and a test set consisting of 3000 samples. We repeat the experiments $K = 10$ times and report the average error rate with standard deviation. To determine the model architecture in each case we took a subset from the training data (3000 and 30,000 for USPS and MNIST respectively), picked randomly, and ran 10-fold cross-validation. We implemented non-linear-NCA and Autoencoders in Python, and we used the scikit-learn (Pedregosa et al., 2011) package to run supervised UMAP. For the rest of the methods, we took the published results for comparison.

Experiment 2: We conducted the second experiment on Letter, Landsat, and Phoneme data sets. For evaluation, we compared CE with the following techniques: supervised neighbor retrieval visualizer (SNeRV) by Venna et al. (2010), multiple relational embedding (MRE) by Memisevic and Hinton (2004), colored maximum variance unfolding (MUHSIC)

Dataset	Network topology	Activation
MNIST	$d \to 1000 \to 500 \to 125 \to 2$	tanh
USPS	$d \to 2000 \to 1000 \to 500 \to 2$	relu
Phoneme	$d \to 250 \to 150 \to 2$	relu
Letter	$d \to 250 \to 150 \to 2$	relu
Landsat	$d \to 250 \to 150 \to 2$	relu
Iris	$d \to 100 \to 2$	relu
Sonar	$d \to 500 \to 250 \to 2$	relu
SUSY	$d \to 500 \to 250 \to 125 \to 2$	relu

Table 1: Network topology used for CE on various data sets. The number d is the input dimension of the network and is data set dependent.

by Song et al. (2007), supervised isomap (S-Isomap) by Cheng et al. (2012), parametric embedding (PE) by Iwata et al. (2007), and neighborhood component analysis (NCA) by Goldberger et al. (2004) and supervised UMAP. Following the experimental setup in Venna et al. (2010), we randomly selected 1500 samples from each data set and ran 10-fold cross-validation ($K = 10$). Except for UMAP and supervised UMAP, we used the published results in (Venna et al., 2010). We picked the architecture (for CE and AE) and other model specific hyper-parameters by running 10-fold internal cross-validation on the training set.

Experiment 3: For the third experiment, we compared the performance of centroid-encoder with supervised PCA (SPCA), and kernel supervised PCA (KSPCA) on the Iris, Sonar, and a subset of USPS data set. Following (Barshan et al., 2011), test error rates were obtained by averaging over $K = 25$ runs on randomly generated 70/30 splits of each data set. For USPS, we randomly picked 1000 cases and used that subset for our experiment, as done by (Barshan et al., 2011). Model specific hyper-parameters are tuned using 10-fold internal cross-validation on the training set. We implemented SPCA and KSPCA in Python.

4.2 Implementation

We have implemented centroid-encoder in PyTorch to run on GPUs. To train CE with an optimal number of epochs, we used 10% of training samples as a validation set in all of our visualization experiments. We measure the generalization error on the validation set after every training epochs. Once the validation error doesn't improve, we stop the training. Finally, we merge the validation set with the training samples and train the model for additional epochs. The model parameters are updated using Adam optimizer (Kingma and Ba, 2014). Table 1 provides the network architecture used in training of CE on various data sets. The implementation is available at: `https://github.com/Tomojit1/Centroid-encoder/tree/master/GPU`. Apart from centroid-encoder, we have implemented autoencoder (AE), non-linear NCA (NNCA), supervised PCA (SPCA) and kernel supervised PCA (KSPCA). These models require the tuning of hyper-parameters as listed in Table 2.

Model	Hyper parameter	Range of Values
CE, AE	*learning_rate*	0.1, 0.01, 0.001, 0.0001, 0.0002, 0.0004, 0.0008
	mini_batch_size	16, 25, 32, 50, 64, 75,128, 256, 512, 1024
	weight_decay	0.001, 0.0001, 0.00001, 0.00002, 0.00004, 0.00008
SUMAP	*n_neighbors*	5, 10, 20, 40, 80
	min_dist	0.0125, 0.05, 0.2, 0.8
KSPCA	*kernel_parameter*(γ)	0.001, 0.01, 0.025, 0.035, 0.05, 0.1, 1.0, 2.5, 3.0, 5.0, 7.5, 10.0

Table 2: Hyper-parameters for different models.

4.3 Quantitative and Visual Analysis

Now we present the results from a comprehensive quantitative and qualitative analysis across these diverse data sets. Note that the primary objective of our assessment is to determine the quality of information obtained from visualization, including class separability, data scatter, and neighborhood structure. Since this assessment is by its very nature subjective, we also employ label information to determine classification rates that provide additional quantitative insight into the data reduction for visualization. Computational expense is also an essential factor, and we will see that this is a primary advantage of CE over other methods when the visualizations and error rates are comparable.

4.3.1 MNIST

Method	Dataset	
	MNIST	USPS
Centroid-encoder	2.61 ± 0.09	2.91 ± 0.31
Supervised UMAP	3.45 ± 0.03	6.17 ± 0.23
NNCA	4.71 ± 0.57	6.58 ± 0.80
Autoencoder	22.04 ± 0.78	$16.49 \pm .91$
dt-MCML	**2.03**	$\mathbf{2.46 \pm 0.35}$
dG-MCML	2.13	3.37 ± 0.18
GerDA	3.2	NA
dt-NCA	3.48	5.11 ± 0.28
dG-NCA	7.95	10.22 ± 0.76
AAE	4.20	NA
pt-SNE	9.90	NA

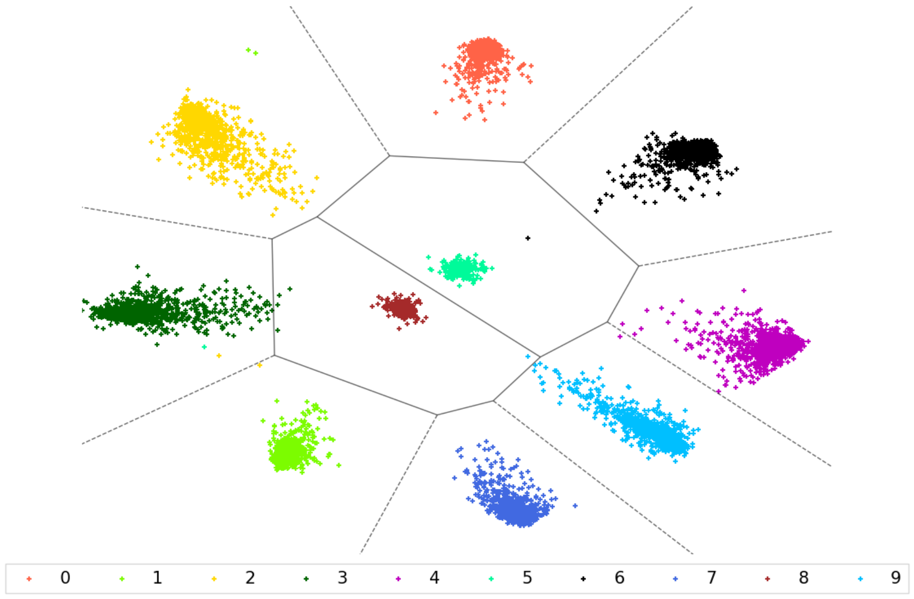
Table 3: Error rates (%) of k-NN (k=5) on the 2D embedded data by various dimensionality reduction techniques trained with pre-training. Results of GerDA and pt-SNE are taken from Stuhlsatz et al. (2012) and van der Maaten (2009) correspondingly. Error rates of the variants of NCA and MCML are reported from Min et al. (2010). The result of AAE is reported from Makhzani et al. (2015). NA indicates that the result was not reported in the original source.

As shown in Tables 3 (results with pre-training) and 4 (results without pre-training), the models with relatively low error rates for the MNIST data amongst our suite of visualization methods are dt-MCML, dG-MCML, centroid-encoder, and HOPE. With pre-training, the error rate of CE is comparable to the top-performing model dt-MCML and superior to NNCA and supervised UMAP by a margin of 2.1% and 0.84%, respectively. Note that methods in the MCML and NCA families require the computation of distance matrix over the data set, making them significantly more expensive than CE, which only requires distance computations between the data of a class and its center. It's noteworthy that the error of CE is lower than the other non-linear variants of NCA: dt-NCA and dG-NCA. We also compare CE with a semi-supervised model called adversarial autoencoders (AAE), which used the label information of some samples to create the two-dimensional embedding. We see that the error rate of CE is better by almost 2%, which is not surprising as CE is a fully supervised method, whereas AAE is semi-supervised. Among all the models, parametric t-SNE (pt-SNE) and autoencoder exhibit the poorest performance in terms of classification error rates on MNIST data are 9.90% and 22.04%, respectively. These high error rates are not surprising given these methods do not use any label information. The prediction error of CE with pre-training is relatively low at 2.6%, as shown in Table 3 behind the more computationally expensive dt-MCML and dG-MCML algorithms. With no pre-training, the numeric performance of centroid-encoder is statistically equivalent to dt-MCML and HOPE. The HOPE algorithm for supervised data visualization is based on the MCML objective function and learns a shallow network for data reduction using label information. Instead of computing the objective on all sample pairs, the authors choose a small set of samples, they refer to as exemplars, to calculate the MCML objective. These exemplars are computed using either k-means or as part of an optimization problem. Note, however, that HOPE requires the user to explore n choose k features, where n is the ambient dimension of the data and k the number of features being used; hence this approach suffers from a combinatorial explosion and is only practical for small k.

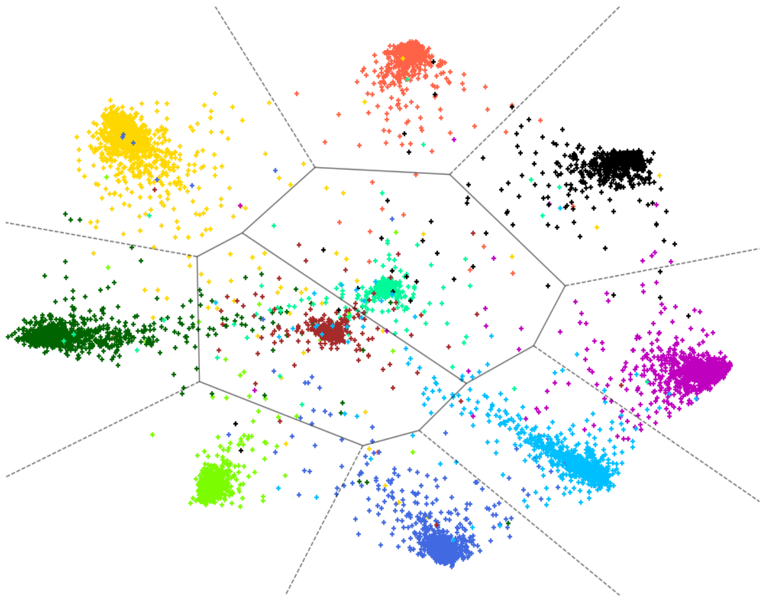
Method	Dataset	
	MNIST	USPS
Centroid-encoder	$\mathbf{3.17 \pm 0.24}$	$\mathbf{2.98 \pm 0.67}$
Autoencoder	21.55 ± 0.47	15.17 ± 0.85
HOPE	3.20	3.03
dt-MCML	3.35	4.07
dt-NCA	3.48	5.11

Table 4: Error rates (%) of k-NN (k=5) on the 2-dimensional data by various techniques trained without pre-training. Error rate of HOPE, dt-NCA and dt-MCML are reported from Min et al. (2017).

While the prediction error is essential as a quantitative measure, the visualizations reveal information not encapsulated in this number. The neighborhood relationships established by the embedding provide potentially valuable insight into the structure of the data set. The two-dimensional centroid-encoder visualization of the MNIST training and test sets are shown in Figure 2. The entire 10,000 test samples are shown in b) while only a subset of the training data set, 1000 digits picked randomly from each class, is shown in a). The separation among the ten classes is easily visible in both training and test data, although there are few overlaps among the categories in test data. Consistent with the low error rate, the majority of the test digits are assigned to the correct Voronoi cells.



(a) Visualuzation of 1000 digits per class from MNIST training set.



(b) Visualization of the 10,000 MNIST test samples.

Figure 2: Voronoi cells in 2D of the MNIST data using centroid-encoder. The network architecture of $784 \to [1000, 500, 125, 2, 125, 500, 1000] \to 784$ is employed to map the data onto the 2D space. The centroid of the training samples mapped to 2D are used to form the Voronoi regions for each digit class.

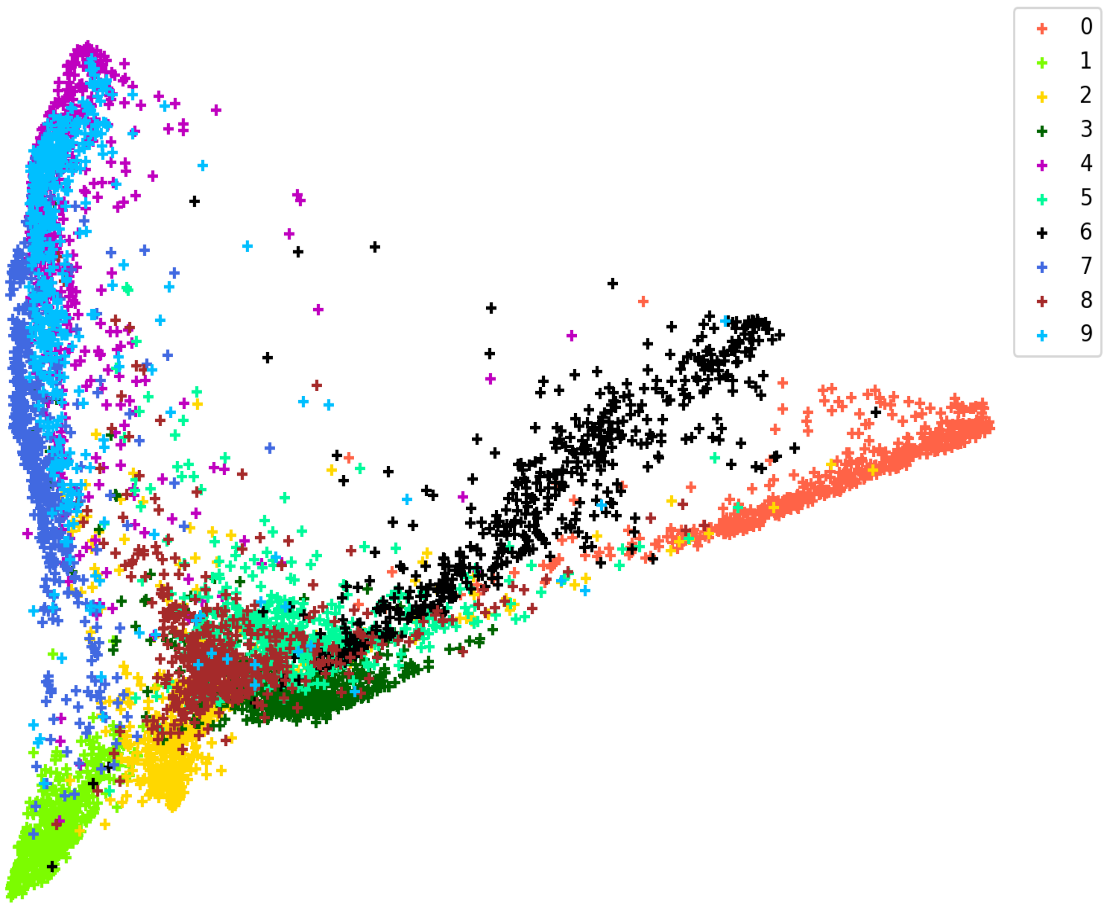


Figure 3: Two dimensional visualization of 10,000 MNIST test digits by Laplacian Eigenmap.

Now let's consider the visualization from a neighborhood relationship perspective. We compare the results with those of Laplacian Eigenmaps (LE), as shown in Figure 3, an unsupervised manifold learning spectral method that solves an optimization problem preserving neighborhood relations (Belkin and Niyogi, 2003). Both CE and LE place digits 5 and 8 as neighbors at the center, allowing us to view all digits as being perturbations of these numbers. Digits 7, 9, and 4 are collapsed to one region; in contrast, CE is built to exploit label information and separates these neighboring digits. LE clumps 7, 9, 4 next to 1 as well, consistent with the CE visualization. The digit 0 neighbors 6 for both CE and LE, but LE significantly overlaps the 6 with the digit 5.

Figure 4 shows the two-dimensional arrangement of 10,000 MNIST test set digits using non-linear NCA, supervised UMAP, UMAP, autoencoder, and t-SNE. It is apparent that CE is more similar to LE in terms of the relative positioning of the digits in 2D than these other visualization methods. There are some similarities across all the models, e.g., 0 and 6 are neighbors for all methods (except for SUMAP), as are the digits 4, 9, and 7. The separation among the different digit groups more prominent in CE than all the methods save for SUMAP, but as seen above, quantitatively, SUMAP has a higher error rate than CE. We note that unsupervised UMAP, as well as t-SNE, both agree with CE that the digits 5 and 8 should be in the center. However, in contrast, CE provides a mapping that can be applied to streaming data.

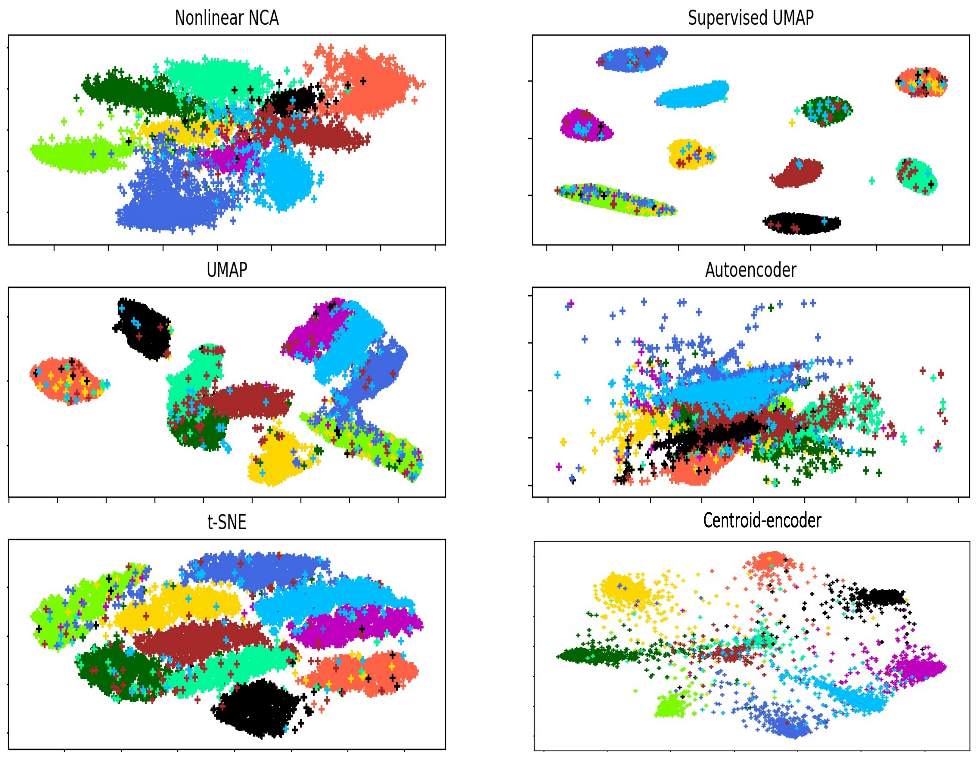


Figure 4: A comparison of the visualizations of 10,000 MNIST test digits by different methods.

4.3.2 USPS

On USPS, centroid-encoder without pre-training has a prediction error slightly lower than HOPE but outperformed dt-MCML and dt-NCA by the margins of 1.09% and 2.12%, respectively, see Table 4. Without pre-training, the centroid-encoder performed better than NNCA with pre-training and supervised-UMAP on both MNIST and USPS datasets.

On the USPS data, the top three models with pre-training are dt-MCML, centroid-encoder, and dG-MCML with the error rates of 2.46%, 2.91%, and 3.37%, respectively. The error rate of centroid-encoder is better than NNCA and supervised-UMAP by a margin of 3.67% and 3.26%, respectively. Like in MNIST, dt-NCA and dG-NCA performed relatively poorly compared to centroid-encoder. Again, autoencoder performed the worst among all the models.

In Figure 5, we show the two-dimensional visualization of 3,000 test samples using non-linear NCA, supervised UMAP, centroid-encoder, and autoencoder. Like MNIST, the neighborhoods of CE and the other methods share many similarities. Digits 4, 9, and 7 are still neighbors in all the techniques but now sit more centrally. Digits 8 and 6 are now consistently neighbors across all methods. The within-class scatter of each digit is again the highest for autoencoder since it is unsupervised. Non-linear NCA also has considerable variance across all the digit classes. Supervised UMAP has tighter clumping of the classes

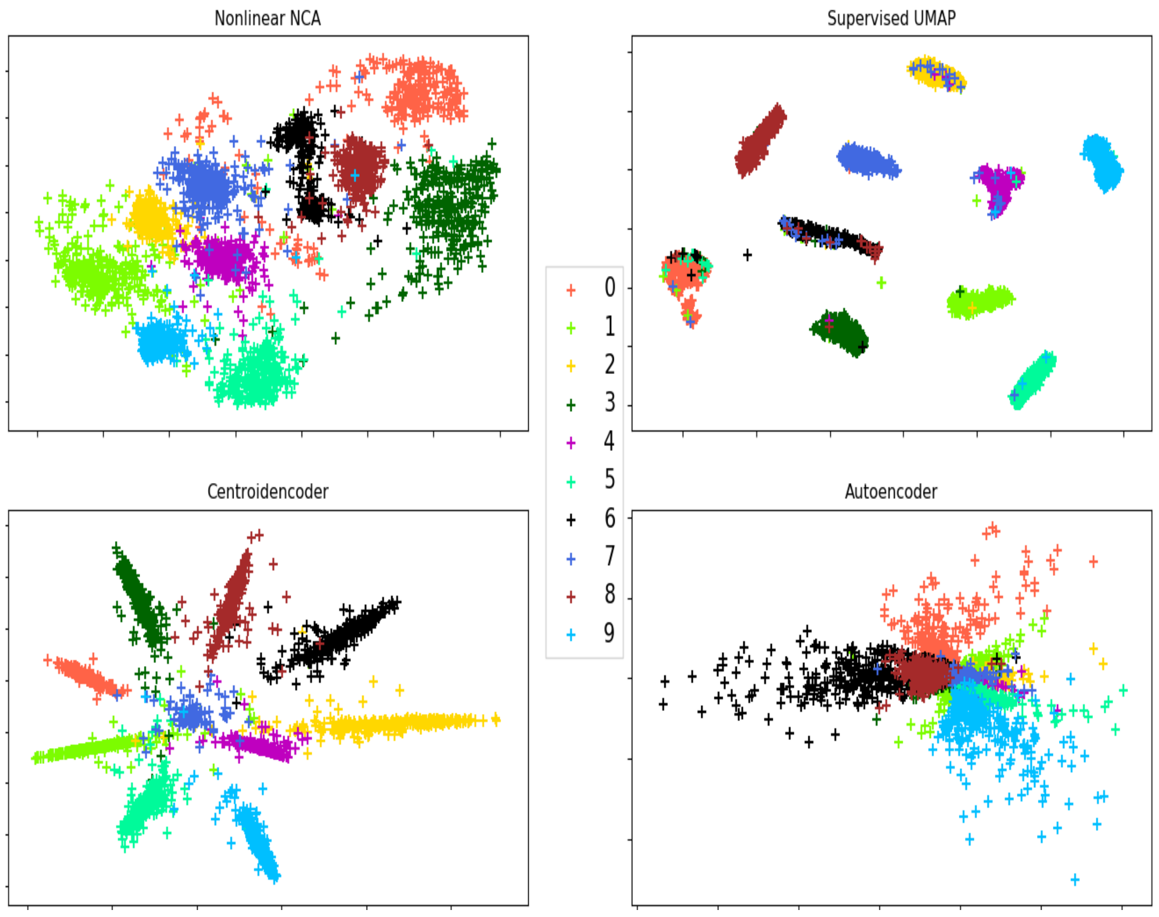


Figure 5: Two dimensional plot of 3000 USPS test digits by different dimensionality reduction methods.

– apparently an integral feature of SUMAP as well as UMAP. However, supervised UMAP has some misplaced digits compared to CE, which is evident from the error rate in Table 3. These observations are consistent with MNIST visualizations. SUMAP emphasizes the importance of local structure over the global structure of the data. In contrast to CE, Autoencoder doesn't provide any meaningful information about the neighborhood structure.

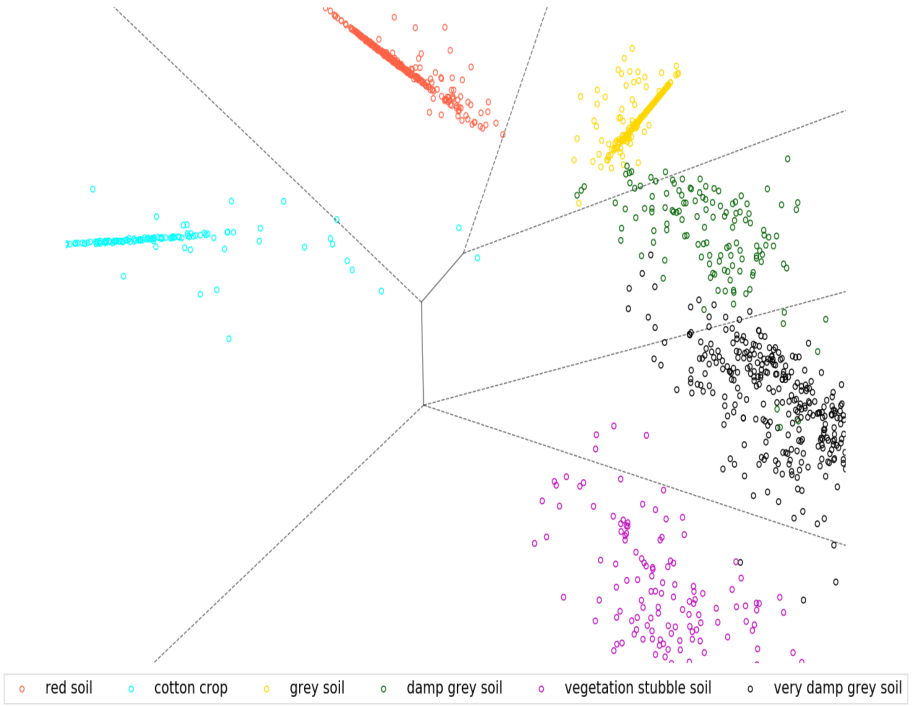
4.3.3 Letter, Landsat and Phoneme Data Sets

Here we compare centroid-encoder with another suite of supervised methods including, SNeRV, PE, S-Isomap, MUHSIC, MRE, and NCA. Table 5 shows the results on three benchmarking data sets, including UCI Letter, Landsat, and Phoneme data set. We included UMAP for additional comparison. CE produces the smallest prediction error on Landsat and Phoneme data sets and is ranked second on the Letter data set. On Landsat, the top three models are CE, SUMAP and UMAP. On the Phoneme data, the centroid-encoder outperforms the second-best model, which is SUMAP, by a small margin of 0.09%. Notably, the variance of errors of centroid-encoder (1.33) is better than the SUMAP (2.84). On the Letter dataset, the centroid-encoder achieves the error rate of 23.82%, which is the second-best model after SNeRV ($\lambda = 0.1$), although the variance of the result of centroid-encoder is better than SNeRV by a margin of 1.47. Surprisingly the performance of SUMAP and UMAP degrade significantly on this particular data set. It's also noteworthy that the variance of errors for centroid-encoder is the lowest *in every case.*

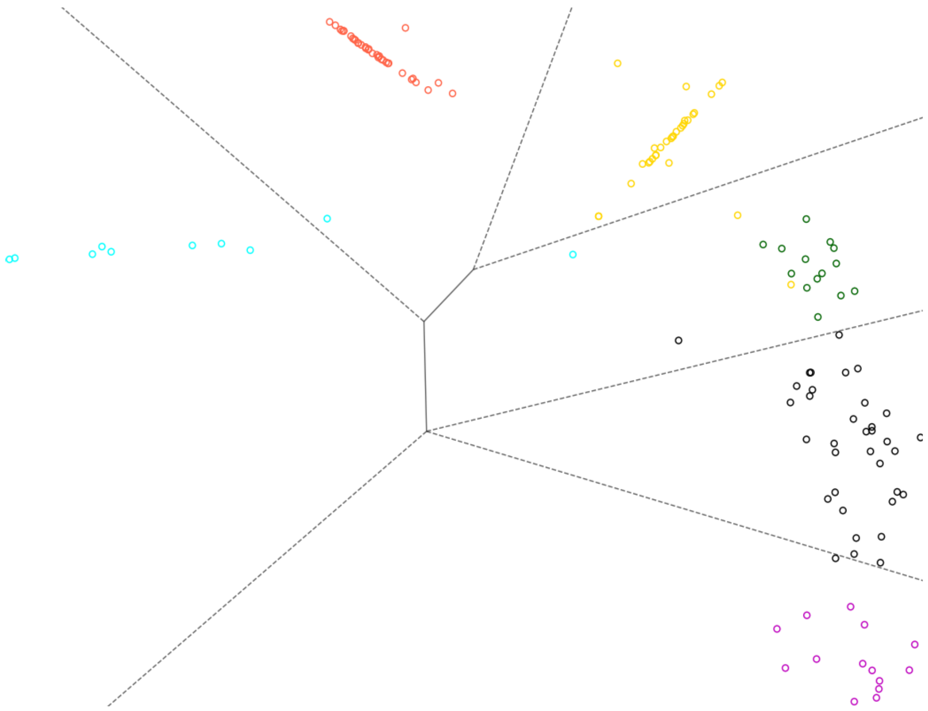
Method	Dataset		
	Letter	Landsat	Phoneme
CE	23.82 ± 3.13	$\mathbf{8.97 \pm 2.55}$	$\mathbf{7.52 \pm 1.33}$
Supervised UMAP	33.57 ± 5.29	9.68 ± 3.12	7.61 ± 2.84
UMAP	39.15 ± 5.51	11.89 ± 2.83	8.47 ± 3.08
SNeRV $\lambda = 0.1$	$\mathbf{22.96 \pm 4.6}$	14.34 ± 7.38	9.43 ± 7.79
SNeRV $\lambda = 0.3$	24.59 ± 4.6	13.93 ± 6.97	9.02 ± 7.38
PE	31.15 ± 4.92	14.75 ± 8.20	9.84 ± 6.15
S-Isomap	31.97 ± 7.38	15.16 ± 9.02	8.61 ± 5.74
NCA	62.30 ± 5.74	15.57 ± 7.38	20.49 ± 5.73
MUHSIC	79.51 ± 4.92	15.37 ± 4.10	14.75 ± 4.1
MRE	90.98 ± 7.38	53.28 ± 34.2	45.08 ± 18.03

Table 5: Classification error (%) of k-NN (k=5) on the 2D embedded data by different supervised embedding methods on Letter, Landsat and Phoneme data. Average misclassifications over ten-fold cross-validation are shown along with standard deviation. Results of methods other than centroid-encoder, UMAP and SUMAP are reported from Venna et al. (2010).

The visualization of the two-dimensional embedding using centroid-encoder on Landsat is shown in Figure 6. We might conclude that the water content in the samples is causing scatter in the 2D plots and decreases as we proceed counter-clockwise. The test samples encode essentially the same structure. In Figure 7, we show the visualization using SUMAP. We see tighter clusters, but there is no difference between the damp and non-damp soil types. The neighborhood relationships, demonstrated by Laplacian Eigemaps, also show this transition by dampness consistent with CE. Damp and very damp soils are, however, not neighbors using SUMAP, making the visualization less informative.

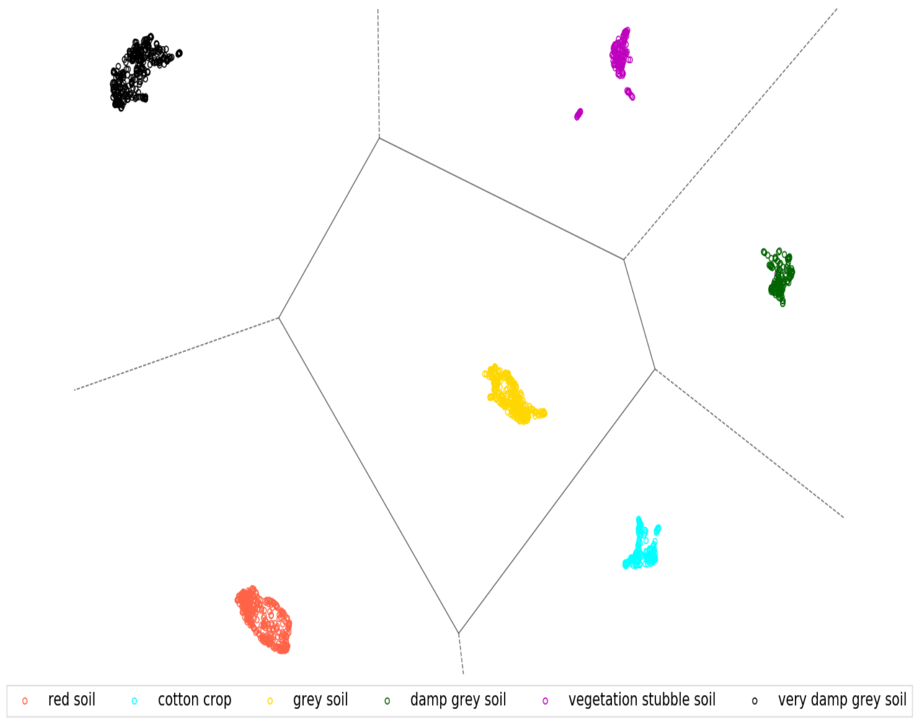


(a) Two-dimensional plot of Landsat training samples.

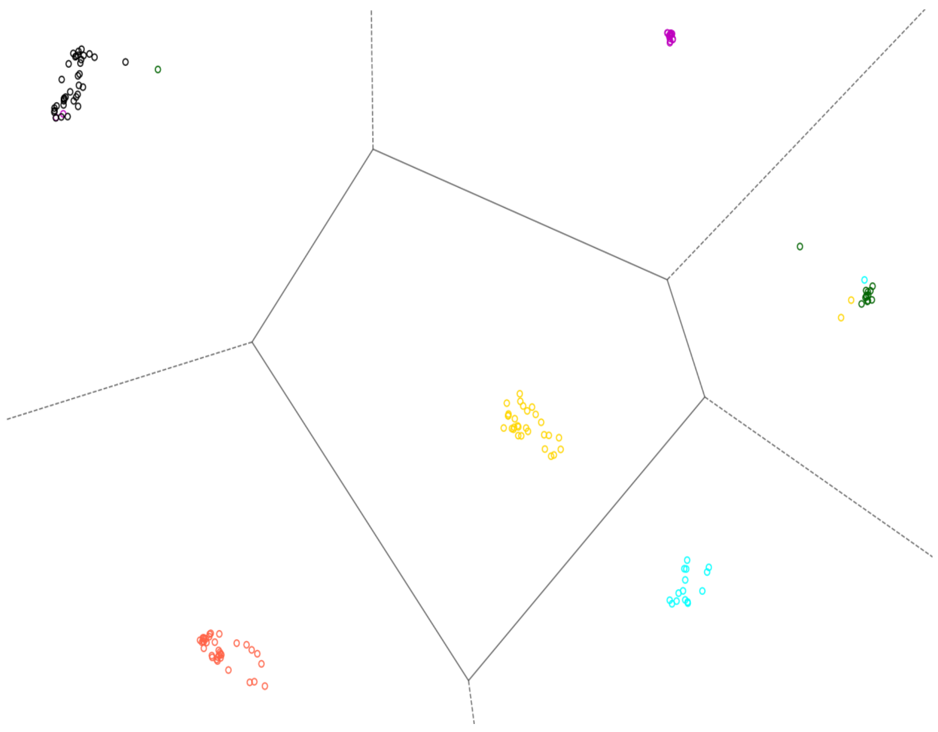


(b) Two-dimensional plot of Landsat test samples.

Figure 6: Voronoi cells of two-dimensional Landsat data using a $16 \rightarrow [250, 150, 2, 150, 250] \rightarrow 16$ centroid-encoder. Voronoi regions for each soil type are formed from the training set.



(a) Two-dimensional plot of Landsat training samples.



(b) Two-dimensional plot of Landsat test samples.

Figure 7: Voronoi cells of two-dimensional Landsat data using Supervised UMAP. Voronoi regions for each soil type are formed from the training set.

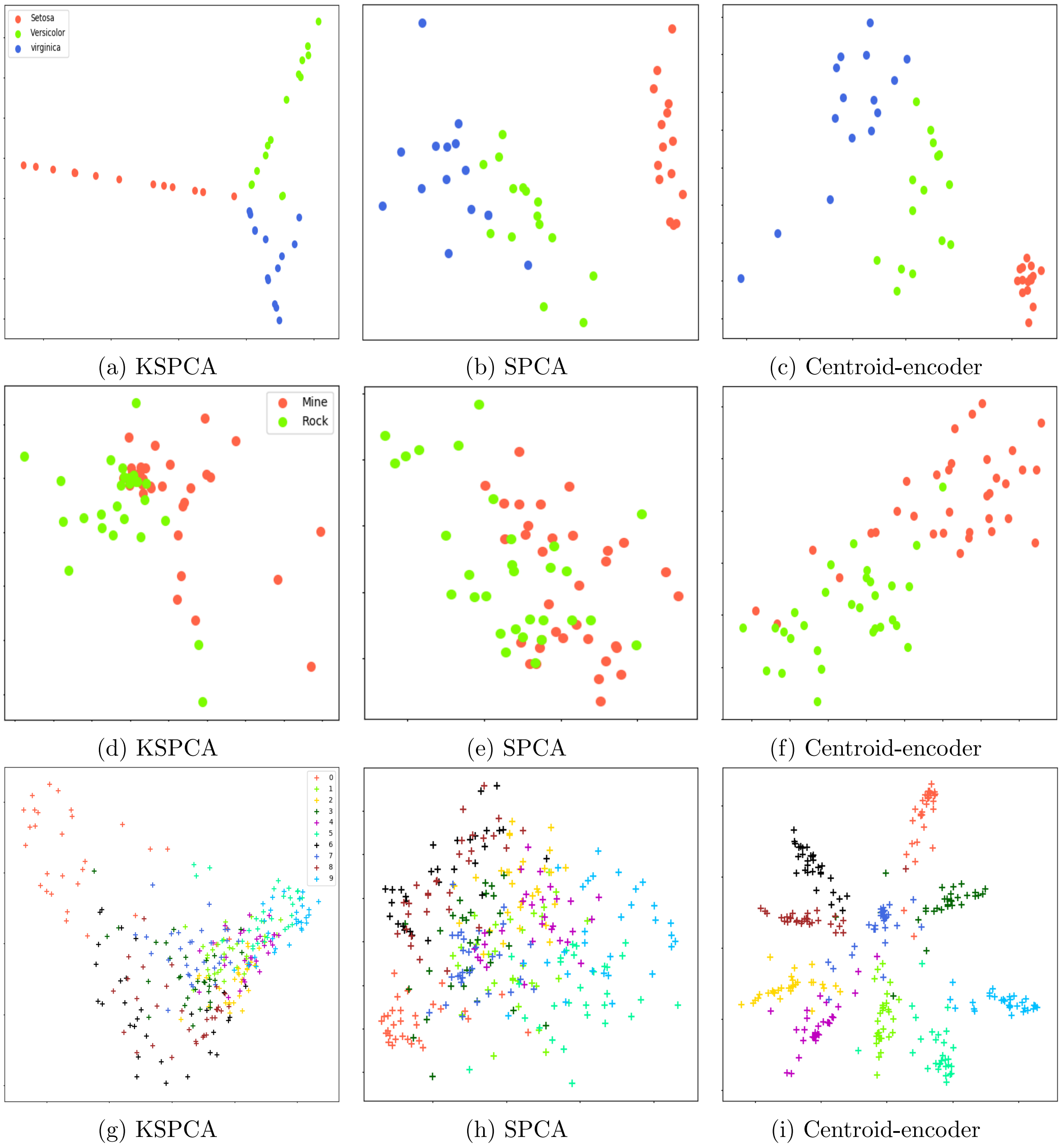


Figure 8: Visualization of Iris (top row), Sonar (middle row) and USPS (bottom row) data using different dimensionality reduction techniques.

Method	Dataset		
	IRIS	Sonar	USPS
Centroid-encoder	**3.29 ± 2.10**	**14.24 ± 2.99**	**12.16 ± 2.01**
KSPCA	5.16 ± 4.56	21.06 ± 5.38	51.63 ± 2.11
SPCA	5.24 ± 2.23	32.75 ± 6.78	54.18 ± 0.66

Table 6: k-NN ($k = 5$) error (%) on the 2D embedded data by different supervised embedding methods on Iris, Sonar and USPS data. Note that given the limitations of KSPCA we have restricted the USPS data set to have 1000 total samples.

4.3.4 Iris, Sonar and USPS (revisited) Data Sets

In this experiment, we compare supervised PCA and kernel supervised PCA to centroid-encoder. Table 6 contains the classification error using a 5-NN classifier, showing that centroid-encoder outperforms SPCA and KSPCA by considerable margins by this objective measure. In the top row of Figure 8, we present the two-dimensional visualization of Iris test data. Among the three methods, centroid-encoder and KSPCA produce better separation compared to SPCA. SPCA can't separate the Iris plants Versicolor and Virginica in 2D space. The dispersion of the three Iris categories is relatively higher in KSPCA and SPCA compared to centroid-encoder. Surprisingly in KSPCA, three classes are mapped on three separate lines.

The two-dimensional visualization of Sonar test samples is shown in the middle row of Figure 8. None of the methods can completely separate the two classes. The embedding of KSPCA is slightly better than SPCA, which doesn't separate the data at all. In centroid-encoder, the two classes are grouped relatively far away from each other. Most of the rock samples are mapped at the bottom-left of the plot, whereas the mine samples are projected at the top-right corner.

The visualization of USPS test data is shown at the bottom of Figure 8. Both SPCA and KSPCA are unable to separate the ten classes, although KSPCA separates digit 0s from the rest of the samples. In contrast, the centroid-encoder puts the ten digit-classes in 2-D space without much overlap. Qualitatively, the embedding of centroid-encoder is better than the other two.

4.4 Visualization and Classification of SUSY

The high-energy supersymmetry particle data (SUSY), is a large data set with 4,500,000 training points and 0.5 million test points (Baldi et al., 2014). Here we demonstrate that CE is effective at visualizing larger data sets; taken as a whole this data set is too large for many of the other methods compared here such Laplacian Eigenmaps, t-SNE, KSPCA, SPCA. The sklearn implementation of UMAP required 9 hours 12 minutes to complete on this data set.

In Figure 9, we show the 2-dimensional embedding of SUSY test samples using centroid-encoder, supervised UMAP and PCA. Unlike previous exmaples in this paper, here we used a multi-center per class approach for centroid-encoder. For both the signal and background classes two centers were determined using standard k-Means algorithm (Lloyd, 1982; Mac-

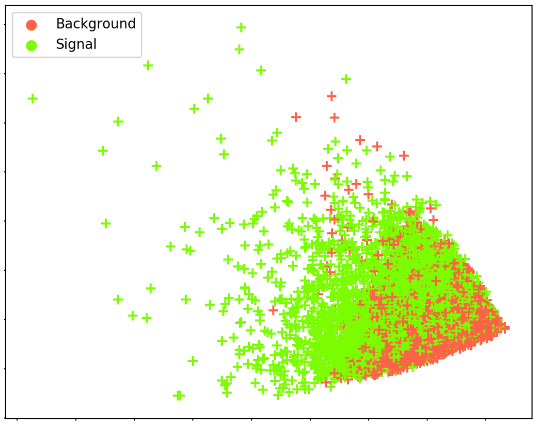
Method	Dim. of classification space	BSR	AUC
CE Classifier	2	$\mathbf{80.41 \pm 0.02}$	0.875 ± 0.002
Baldi et al. (2014)	300	NA	**0.879**
Le et al. (2018)	not reported	77.79 ± 0.02	NA

Table 7: Balanced success rate (BSR) and area under the curve (AUC) measure of different methods on SUSY data sets. NA indicates result not reported.

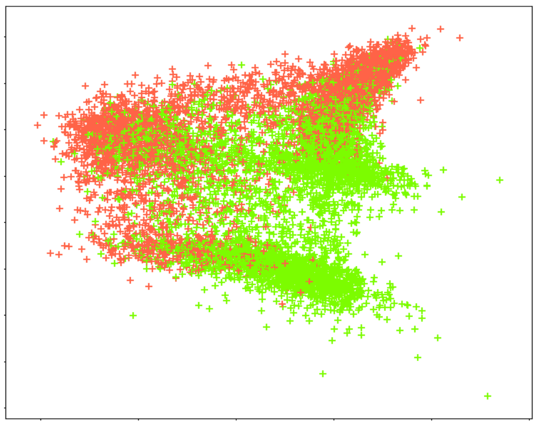
Queen et al., 1967). Now the CE algorithm is applied as before. We observe that the data is not completely separable in two-dimensional space, as evident from the visualization of each model. PCA and centroid-encoder put the two classes close together, although centroid-encoder split the two classes better than PCA. On the other hand, SUMAP produces three distinct clusters: one for the signals for two for the backgrounds. Notice the overlap of the samples from signal to the background.

To quantify the quality of the embedding of these models, we again ran k-NN classification on the two-dimensional space. We vary the number of nearest neighbors (hyper-parameter k) to investigate the robustness of the embedding as we transition from local to more global structure. We see that the accuracies remain almost the same for supervised UMAP as we increase the value of k. On the other hand, the classification improves for centroid-encoder as k increases. The accuracy starts to saturate after $k = 50$. The low classification accuracy of PCA is expected as it's an unsupervised model. It's noteworthy that the classification performance using the embedding of centroid-encoder is better than supervised UMAP for each k. The higher classification accuracies for the small and large values of k indicate that the embedding of centroid-encoder is robust than supervised UMAP. From the neighborhood perspective, we can say that the centroid-encoder better preserves the neighborhood of the data both from local to more global sense.

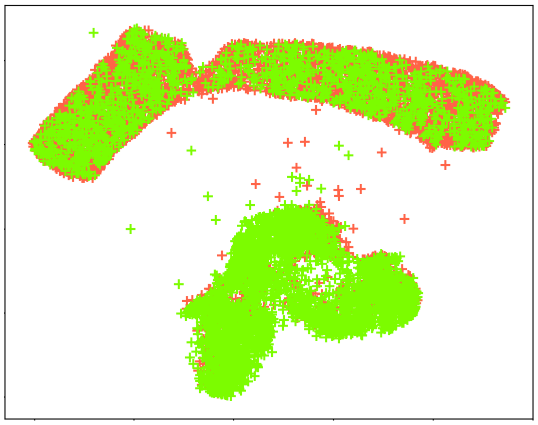
To illustrate how CE can be used as a classifier, we train a one-hot encoded neural network on the visualized data. In this architecture the decoder is discarded and a softmax layer with one-hot-encoding is attached with the encoder. The network is further trained to learn the class label. In this experiment a bottleneck architecture with two nodes in the bottleneck layer is used; see Table 1. The resulting balanced success rate (BSR) of CE is better than the supervised autoencoder model of Le et al. (2018) by a margin of 2.65. The difference of AUC score between our model to Baldi et al. (2014) is not significant. We should note that we performed the classification on the 2-dimensional space, whereas Baldi et al. (2014) did the classification on 300-dimensional space; see Table 7.



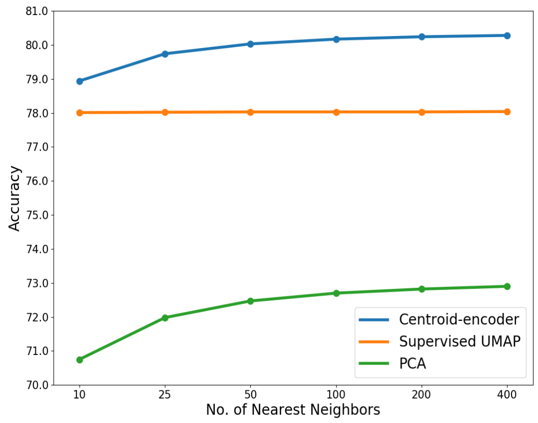
(a) PCA projection of the test data set.



(b) CE reduction of the test data set.



(c) SUMAP representation of the test data set.



(d) k-NN classification accuracy.

Figure 9: Visualization of SUSY data set using PCA, centroid-encoder and supervised UMAP. We built the models on the training set, and then project the test samples using the trained models. We display a subset of the test set (5,000 samples selected randomly from each class) using the three models. The k-NN classification accuracy is shown in panel (d).

5. Discussion and Analysis of Results

The experimental results in Section 4.3 establish that centroid-encoder performs competitively, i.e., it is the best, or near the best, on a diverse set of data sets. The algorithms that produce similar classification rates require the computation of distance matrices and are hence more expensive than centroid-encoders. Here we analyze the CE algorithm in more detail to discover why this improved performance is perhaps not surprising. This has to do, at least in part, with the fact that centroid-encoder, implementing a nonlinear mapping, is actually able to capture more variance than PCA, and this is particularly important in lower dimensions.

5.1 Variance Plot to Explain the Complexity of Data

PCA captures the maximum variance over the class of orthogonal transformations (Kirby, 2001). The total variance captured as a function of the number of dimensions is a measure of the complexity of a data set. We can use variance to establish, e.g., that the digits 0 and 1 are less complex than the digits 4 and 9. To this end, we use two subsets from the MNIST training set: the first one contains all the digits 0 and 1, and the second one has all the digits 4 and 9. Next, we compute the total variance captured as a function of dimension; see Figure 10. Clearly, the variance is more spread out across the dimensions for the digits 4/9. In Figure 11, we show the visualization of these two subsets using PCA. We see that the 0/1 samples in subset1 are well separated in two-dimensional space, whereas the data in subset2 are clumped together. This lies in the fact that in low dimension (2D) PCA captures more variance from subset1 (around 41%) as compared to subset2 (about 22%). We can say that subset2 is more complex than subset1 .

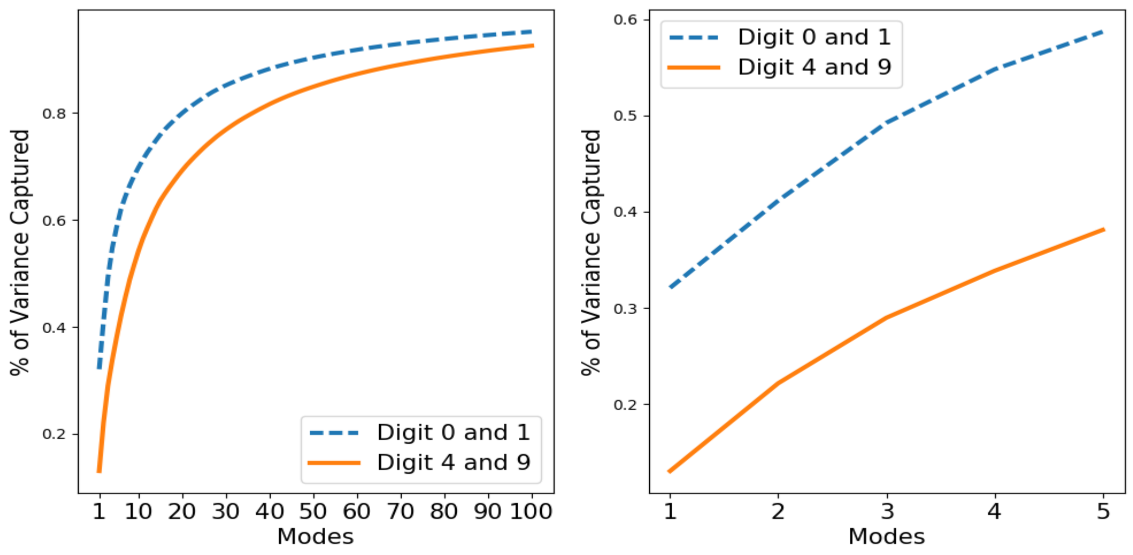
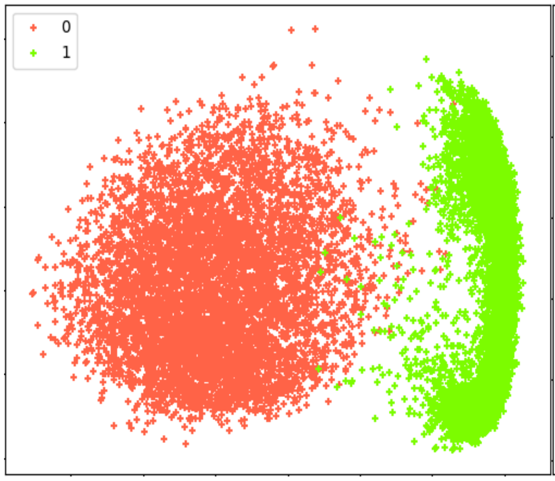
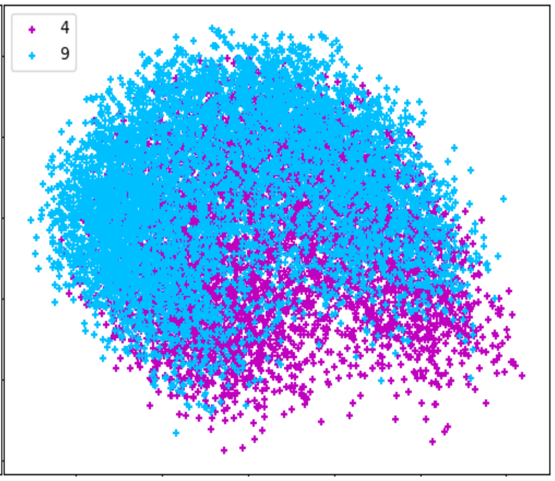


Figure 10: Variance plot using two different subsets (subset1: all the digits 0 and 1, subset2: all the digits 4 and 9) of MNIST data. Left: comparison of variance plot of the two subsets using the first 100 dimensions. Right: a blowup of the plot on the left showing the % of variance captured in low dimensions.



(a) Subset 1: digit 0 and 1



(b) Subset 2: digit 4 and 9

Figure 11: Two dimensional embedding using PCA for subset 1 and subset 2.

5.2 Centroid-encoder and Variance

In this section, we demonstrate how centroid-encoder maps a higher fraction of the variance of the data to lower dimensions. Again, consider the subset of digits 4 and 9 from the experiment above. We use all the digits 4 and 9 from the MNIST test set as a validation set. We trained a centroid-encoder with the architecture $784 \rightarrow [784] \rightarrow 784$ where we used hyperbolic tangent ('tanh') as the activation function. We keep the number of nodes the same in all the three layers. After the training, we pass the training and test samples through the network and capture the hidden activation which we call CE-transformed data. We show the variance plot and the two-dimensional embedding by PCA on the CE-transformed data. Figure 12 compares the variance plot between the original data and the CE-transformed data. In the original data, only 22% of the total variance is captured, whereas about 89% of the total variance is captured in CE-transformed data. It appears that the non-linear transformation of centroid-encoder puts the original high dimensional data in a relatively low-dimensional space. Visualization using PCA of CE-transformed data (both training and test) is shown in Figure 13. This time, PCA separates the classes in two-dimensional space. Comparing Figure 13 with Figure 11b, it appears that the transformation by centroid-encoder is helpful for low-dimensional visualizations.

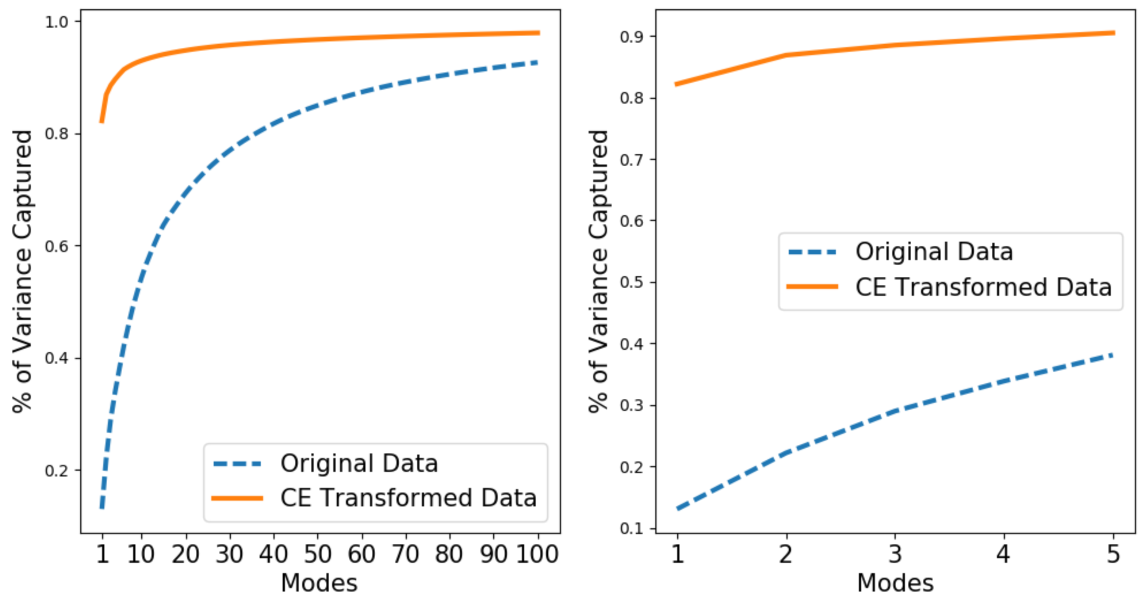
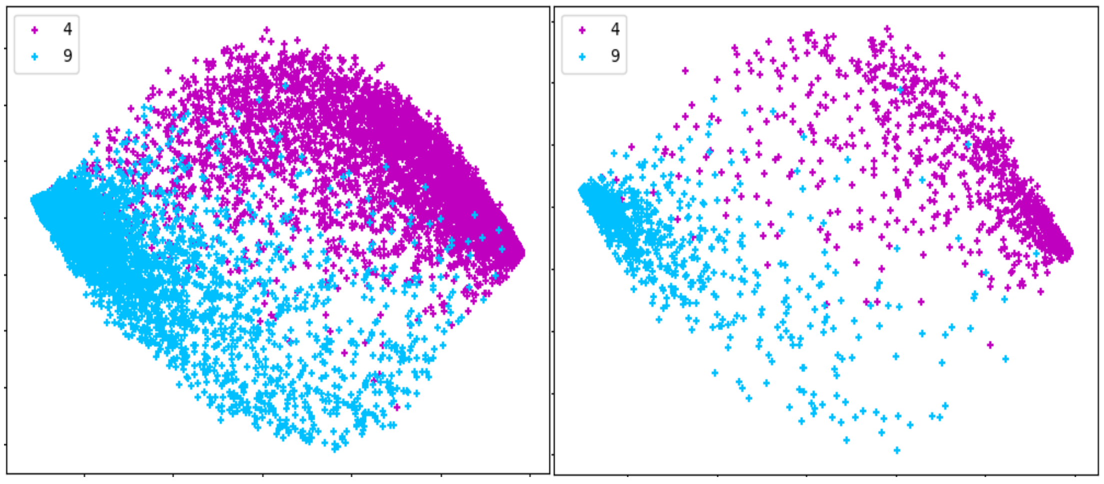


Figure 12: Variance plots for the 4/9 digit data. Left: comparison of variance plot of the original data and CE transformed data using the first 100 dimensions. Right: a blowup of the plot on the left showing the % of variance captured in low dimensions.



(a) Digits 4 and 9 from training set.

(b) Digits 4 and 9 from test set.

Figure 13: Two dimensional embedding using PCA on the CE-transformed data.

5.3 Computational Complexity and Scalability

The computational complexity of the centroid-encoder is effectively the same as the autoencoder. Note, both methods, as is typical of neural network approaches, require the determination of a suitably tuned network architecture. This is a hidden cost for neural networks in general. The advantage of centroid-encoder over many other data visualization techniques is that it is not necessary to compute pairwise distances. If everything else is fixed, we can view centroid-encoder as scaling linearly with the number of data points. Techniques that require distance matrices scale quadratically with the number of data points and can be prohibitively slow as the data set size increases. Methods based on distance matrices include NNCA, dt-NCA, dG-NCA, dt-MCML, and dG-MCML, as well as the spectral methods including MDS, UMAP, Laplacian Eigenmaps, Isomap and their supervised counterparts. The only overhead of centroid-encoder is the calculation of centroids of each class, but this is also linear with the number of classes.

6. Conclusion

In this paper, we presented the centroid-encoder as a new tool for data visualization when class labels are available. CE is well suited to multi-class data visualization tasks for very large data sets, e.g., over a million data points, such as the SUSY analysis presented here. In our experience, it is also beneficial for smaller biological data sets residing in high-dimensions, e.g., 1000 samples with 10,000-50,000 features; this domain-specific application will appear elsewhere. The objective function can also be easily adapted via the addition of penalty terms, such as the addition of a sparsity promoting ℓ_1-norm term. In future work, we will explore the application of this penalty term with CE as a sparse non-linear approach for supervised data reduction that can be applied to the optimal feature extraction problem. The wide variety of examples presented here suggest that CE shares many qualitative performance similarities with techniques such as Laplacian eigenmaps and Fisher Discriminant analysis, but has the advantages associated with autoencoders and deep neural networks including the ability to model very large data sets with highly parameterized deep architectures.

We compared CE to state-of-the-art techniques and showed advantages empirically over other methods. Our examples illustrate that CE captures the topological structure, i.e., neighborhoods of the data, comparable to methods, but with lower cost and generally better prediction error. These experiments include comparisons to the unsupervised Laplacian eigenmaps, UMAP, pt-SNE, autoencoders; the supervised UMAP, SPCA, KSPCA as well as the MCML and NCA class of algorithms. The algorithms that have the most similar prediction errors when compared to CE require the computation of distances between all pairs of points. We demonstrated that the 2D embedding of centroid-encoder produces competitive, if not optimal, prediction errors. We also showed empirically that the model globally captures high statistical variance relative to optimal linear transformations, i.e., more than PCA.

In addition to capturing the global topological structure of the data in low-dimensions, CE exploits data labels to minimize the within-class variance. This improves the localization of the mapping such that points are mapped more faithfully to their Voronoi regions in low-dimensions. CE also provides a mapping model that can be applied to new data without any retraining being necessary. This is in contrast to the spectral methods described here that require the entire training and testing set for computing the embedding eigenvectors. Also, while the experiments run in this paper focus on visualization of labeled data sets, it would be interesting to explore the classification performance of bottleneck dimensions larger than three.

In the future, we plan to extend the CE model to the setting of unsupervised and semi-supervised learning.

Acknowledgments

We would like to thank the referees for their careful reading of the manuscript and thoughtful feedback which led to significant improvements in the organization and clarity of the paper. The authors thank Shannon Stiverson and Eric Kehoe for their feedback on a preliminary version of this paper and Manuchehr Aminian for assisting with the GitHub repository. We would also like to acknowledge support for this research from the National Science Foundation under award NSF-ATD 1830676.

References

Eric Bair, Trevor Hastie, Debashis Paul, and Robert Tibshirani. Prediction by supervised principal components. *Journal of the American Statistical Association*, 101(473): 119–137, 2006. doi: 10.1198/016214505000000628. URL `https://doi.org/10.1198/016214505000000628`.

Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014.

Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recogn.*, 44(7):1357–1371, July 2011. ISSN 0031-3203. doi: 10.1016/j.patcog.2010.12.015. URL `http://dx.doi.org/10.1016/j.patcog.2010.12.015`.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, June 2003. ISSN 0899-7667. doi: 10.1162/089976603321780317. URL `http://dx.doi.org/10.1162/089976603321780317`.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, pages 153–160, Cambridge, MA, USA, 2006. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2976456.2976476`.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, 2007. URL `http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf`.

Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

D.S. Broomhead and M. Kirby. A new approach for dimensionality reduction: Theory and algorithms. *SIAM J. of Applied Mathematics*, 60(6):2114–2142, 2000.

D.S. Broomhead and M. Kirby. The Whitney reduction network: a method for computing autoassociative graphs. *Neural Computation*, 13:2595–2616, 2001.

Jian Cheng, Can Cheng, and Yi-nan Guo. Supervised isomap based on pairwise constraints. In *Proceedings of the 19th International Conference on Neural Information Processing - Volume Part I*, ICONIP'12, pages 447–454, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-34474-9. doi: 10.1007/978-3-642-34475-6_54. URL `http://dx.doi.org/10.1007/978-3-642-34475-6_54`.

Sofya Chepushtanova, Elin Farnell, Eric Kehoe, Michael Kirby, and Henry Kvinge. Dimensionality reduction. In *Data Science for Mathematicians*, pages 291–337. Chapman and Hall/CRC, September 2020.

Vin De Silva and Joshua B Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, Technical report, Stanford University, 2004.

Inderjit S Dhillon, Dharmendra S Modha, and W Scott Spangler. Class visualization of high-dimensional data with applications. *Computational Statistics & Data Analysis*, 41 (1):59–90, 2002.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Willey & Sons, New York, 1973.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188, 1936.

Tomojit Ghosh, Xiaofeng Ma, and Michael Kirby. New tools for the visualization of biological pathways. *Methods*, 132:26 – 33, 2018. ISSN 1046-2023. doi: https://doi.org/10.1016/j.ymeth.2017.09.006. URL `http://www.sciencedirect.com/science/article/pii/S1046202317300439`. Comparison and Visualization Methods for High-Dimensional Biological Data.

Amir Globerson and Sam T Roweis. Metric learning by collapsing classes. In *Advances in neural information processing systems*, pages 451–458, 2006.

Jacob Goldberger, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 513–520, Cambridge, MA, USA, 2004. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2976040.2976105`.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Geoffrey E Hinton and Sam T. Roweis. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 857–864. MIT Press, 2003. URL `http://papers.nips.cc/paper/2276-stochastic-neighbor-embedding.pdf`.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, pages 3–10, 1994.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006. ISSN 0899-7667. doi: 10.1162/neco.2006.18.7.1527. URL `http://dx.doi.org/10.1162/neco.2006.18.7.1527`.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, September, 1933.

Tomoharu Iwata, Kazumi Saito, Naonori Ueda, Sean Stromsten, Thomas L. Griffiths, and Joshua B. Tenenbaum. Parametric embedding for class visualization. *Neural Comput.*, 19(9):2536–2556, September 2007. ISSN 0899-7667. doi: 10.1162/neco.2007.19.9.2536. URL `http://dx.doi.org/10.1162/neco.2007.19.9.2536`.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

I.T. Jolliffe. *Principal Component Analysis.* Springer, New York, 1986.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL `http://arxiv.org/abs/1412.6980`.

M. Kirby. *Geometric Data Analysis: An Empirical Approach to Dimensionality Reduction and the Study of Patterns.* Wiley, 2001.

T. Kohonen. Self-organized formation of topologically correct feature maps. *Biol. Cybern.*, 43:59, 1982.

T. Kohonen. Adaptive, associative, and self-organizing functions in neural computing. *Applied Optics*, 26(23):4910–4918, 1987.

Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991a. ISSN 1547-5905. doi: 10.1002/aic.690370209. URL `http://dx.doi.org/10.1002/aic.690370209`.

Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE J.*, 37(2):233–243, 1991b.

Mark A. Kramer. Autoassociative neural networks. *Comput. Chem. Engng.*, 16(4):313–328, 1992.

Lei Le, Andrew Patterson, and Martha White. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. *Advances in neural information processing systems*, 31:107–117, 2018.

Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantization design. *IEEE transactions on Communications*, 28(1):84–95, January 1980.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

Xiaofeng Ma, Michael Kirby, and Chris Peterson. Self-organizing mappings on the flag manifold. In *International Workshop on Self-Organizing Maps*, pages 13–22. Springer, June 2019.

James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Roland Memisevic and Geoffrey Hinton. Multiple relational embedding. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 913–920, Cambridge, MA, USA, 2004. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2976040.2976155`.

Martin R Min, Laurens Maaten, Zineng Yuan, Anthony J Bonner, and Zhaolei Zhang. Deep supervised t-distributed embedding. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 791–798, 2010.

Martin Renqiang Min, Hongyu Guo, and Dongjin Song. Exemplar-centered supervised shallow parametric data embedding. *arXiv preprint arXiv:1702.06602*, 2017.

E. Oja. Data compression, feature extraction, and autoassociation in feedforward neural networks. In T. Kohonen, K. Mäkisara., O. Simula, and J. Kangas, editors, *Artificial Neural Networks*, pages 737–745, NY, 1991. Elsevier Science.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *Phil. Mag. S.*, 2(11):559–572, 1901.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, November 2011. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1953048.2078195`.

B. Raducanu and F. Dornaika. A supervised non-linear dimensionality reduction approach for manifold learning. *Pattern Recogn.*, 45(6):2432–2444, June 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2011.12.006. URL `http://dx.doi.org/10.1016/j.patcog.2011.12.006`.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

Ruslan Salakhutdinov and Geoff Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Artificial Intelligence and Statistics*, pages 412–419, 2007.

Le Song, Alexander J. Smola, Karsten M. Borgwardt, and Arthur Gretton. Colored maximum variance unfolding. In *NIPS*, 2007.

Andre Stuhlsatz, Jens Lippel, and Thomas Zielke. Feature extraction with deep neural networks by a generalized discriminant analysis. *IEEE transactions on neural networks and learning systems*, 23(4):596–608, 2012.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. ISSN 0036-8075. doi: 10.1126/science.290.5500.2319. URL `http://science.sciencemag.org/content/290/5500/2319`.

W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17: 401–419, 1952.

Laurens van der Maaten. Learning a parametric embedding by preserving local structure. In *AISTATS*, volume 5 of *JMLR Proceedings*, pages 384–391. JMLR.org, 2009.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL `http://www.jmlr.org/papers/v9/vandermaaten08a.html`.

Jarkko Venna, Jaakko Peltonen, Kristian Nybo, Helena Aidos, and Samuel Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *J. Mach. Learn. Res.*, 11:451–490, March 2010. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1756006.1756019`.

S. Watanabe. Karhunen–Loève expansion and factor analysis. In *Trans. 4th. Prague Conf. on Inf. Theory, Statist. Decision Functions, and Random Proc.*, pages 635–660, Prague, 1965.

Killan Q. Weinberger and Lawrence K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, pages 1683–1686. AAAI Press, 2006. ISBN 978-1-57735-281-5. URL `http://dl.acm.org/citation.cfm?id=1597348.1597471`.

P.J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD dissertation, Harvard University, August 1974.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

Shi-qing Zhang. Enhanced supervised locally linear embedding. *Pattern Recogn. Lett.*, 30 (13):1208–1218, October 2009. ISSN 0167-8655. doi: 10.1016/j.patrec.2009.05.011. URL `http://dx.doi.org/10.1016/j.patrec.2009.05.011`.