CSE 519 PROJECT PROPOSAL
Evaluating Dimensionality Reduction Methods

Table of Contents

Abstract

The dimensionality reduction technique is the process of reducing the number of variables of a high-dimensional dataset, which on reduction preserve the majority of information from the original dataset. In today's world, there is an abundant flow of data: from phones, credit cards, home appliances, sensor equipped buildings, vehicles, big factories and other enormous number of sources. With this data revolution, the classical statistical analysis as well as making quick decisions based on these data, becomes a new challenge.

The goal is shifted from statistical analysis to new ways of linking datasets with generation of graphical insights. As the ever-increasing data becomes more and more complex, the curse of dimensionality, which forbids to reveal hidden insights in low dimension, hinders the businesses as well as new scientific discoveries. On one hand, the statistical or Machine learning algorithms started to fail due to high variance; while big data has also introduced processing & storage barriers. Dimensionality reduction techniques can sufficiently bring down the higher dimensional data to lower dimensional representations without significant loss of latent information. In this evaluation, we will try to analyze the inherent problems, suitable reduction techniques, the challenges and some recent developments in this area.

Background Research

As the data is increasing, it becomes more difficult to process the real time data as it is incomplete, inconsistent and unorganized. The main challenge is to find the transformation to intrinsic dimension with minimal loss while keeping significance intact. These dimensionality reduction techniques are applied in latent knowledge discovery, feature analysis, machine learning algorithms and visualization. Historically, Linear dimensionality techniques were generally developed in the earlier times, where PCA (**P**rincipal **C**omponent **A**nalysis) was among the most popular ones .But, **PCA** does not work in those cases in uncorrelated data.

These linear techniques transform the data from high to low dimension via linear combination of variables (derived variables). But these linear techniques only work when data lies in linear space, which doesn't cover the majority of problems which are complex and nonlinear. Also, linear dimensionality reduction techniques are not as capable as non-linear methods to product optimal low-dimensional representations when data points in the input space are arranged along highly curved surfaces. To overcome these shortcomings of linear dimensionality reduction methods, non-linear methods were proposed. Among the early invented non-linear methods, MDS and Isomap were popular.

Dimensionality reduction approaches can be divided into feature selection and feature extraction. Feature selection methods try to find a subset of the input attributes. It can be done in the following ways:

1. The *filter* strategy,
2. The *wrapper* strategy, and
3. The *embedded* strategy

Feature extraction transforms the data in the high dimension space to fewer dimensions.
Feature extraction transformation can be linear or nonlinear.

1. Linear Dimensionality techniques: Many methods have been proposed that produces compressed data that preserves some features of interest in the data. Some of the linear techniques are:

    - **PCA** (Principal Component Analysis)
    - **LDA** (Linear Discriminant Analysis)

2. Non-Linear Dimensionality techniques: These can be divided into three categories:

    a. Distance preservation:
    - **MDS** (Multidimensional Scaling)
    - **Diffusion Map**
    - **Isomap**
    - **UMAP** (Uniform Manifold Approximation and Projection)
    - **t-SNE** (T-distributed Stochastic Neighbor Embedding

    b. Kernel-based:
    - **Kernel PCA**

    c. Neural Networks:
    - **Autoencoder**

There are many research papers that give a comparative study about the above-mentioned techniques. Like, for example,

***Laurens van der Maaten* et. al.** presents a review of various linear and non-linear dimensionality reduction techniques. It shows a systematic comparison between various these techniques and how non-linear helps in overcoming the limitations of linear traditional techniques such as PCA. It also evaluates the performance 1 linear (PCA) and 12 non-linear techniques, Kernel PCA, Isomap, diffusion maps are some among those and in the final section, it points out the weaknesses and the area of improvement in various techniques.

***G. E. Hinton* et. al.** shows that auto encoder is a better dimensionality technique than principal component analysis, which was a result from breakthrough of use of Neural Networks in practical dataset. While PCA is a linear method, autoencoder is a non-linear method which uses multilayer encoder network initialized with some random weights and with each layer the weights get autocorrected via back propagation in such a way that the deep neural network is able to create the same output as input.

Popular Reduction Techniques

UMAP

Umap is a novel manifold learning technique for dimensionality reduction. Umap is based on Riemannian geometry and algebraic topology. Umap is competitive with t-SNE in visualization quality and preserves more global structure compared to t-SNE. UMAP can be thought of a two-step process. In the first step, a particular weighted k-neighbor graph is constructed. In the second phase, a low-dimensional layout of the graph is computed. Umap performs better at preserving global aspects of the high dimensional data as compared to t-SNE. It uses binary-cross entropy as a cost function instead of the KL divergence used in t-SNE. This leads to a huge change in the global data preservation. UMAP does not use normalization for high dimensional as well as low dimensional probabilities. Umap somewhat uses the same steps as t-SNE but with some improvements and absence of normalization.

Properties:

1. Umap, unlike t-SNE, does not need pre-dimensionality reduction for converting high-dimensional data.
2. UMAP can be used for supervised and semi-supervised reductions.

t-SNE

T-distributed Stochastic Neighbor Embedding(t-SNE) is a non-linear dimensionality reduction technique used to embed high-dimensional data into low dimensions for visualization. t-SNE, uses t-distribution, to compute similarity between two points. t-SNE works by first constructing the probability distribution for different pairs of objects in high dimensional data to find points that can be similar. After describing the probability function for all pairs of high-dimensional objects, t-SNE maps these into lower dimensional space. It does so by randomly plotting the objects in the lower dimension space, re-computing the probability function for all pairs of objects and then using the earlier probabilities to group objects in the lower-dimensional space. t-SNE has over 10 hyper parameters that can be used for optimization. Some of these include perplexity, number of iterations, learning rate etc.

Properties:

1. t-SNE works on neighborhood distances and preserve local structure while also revealing presence of global features like clusters
2. t-SNE is usually used to embed high-dimensional data into 2-3 dimensions i.e. for visualization purposes. Thus, it is not assured if t-SNE is going to be useful as a general dimension reduction technique,
3. t-SNE requires pre-dimensionality reduction before actually applying the algorithm and consumes high memory for its computations

Autoencoder

Autoencoder is a data compression algorithm, that, by using neural networks, is able to learn how to compress data into small bits of data and then using that regenerate the original representation of the input. There are two main components of the autoencoder, namely, down-sampling and up-sampling. These two steps are used together to efficiently obtain the representation of data from which the original representation is reconstructed, without losing much of the data. The down sampling process basically tries to represent the data in the most effective form by extracting the most prominent features from the input data. The up-sampling step is exactly opposite where in this step the autoencoder tries to read features instead of finding them. It reads features from the compressed data and tries to extract images from them.

Properties:

1. Autoencoders are data-specific, which means they will only be able to compress data similar to the trained data, lossy meaning that there will be loss of some information in the steps and learn automatically from the training data.
2. Autoencoders are neural networks, thus require a high amount of data to train.
3. Autoencoders can only be used to compress data that is similar to the data it has trained on. This limits the practical applications of Autoencoders

Dataset

1. **MNIST**
   The MNIST dataset of database of handwritten digits, published by *LeCun*, which consists of a training set of labelled 60,000 examples with 10,000 test data. Each of the image in the dataset is of shape of 28x28 and grayscale color space with 10 labels representing digits from 0-9.
   Example:

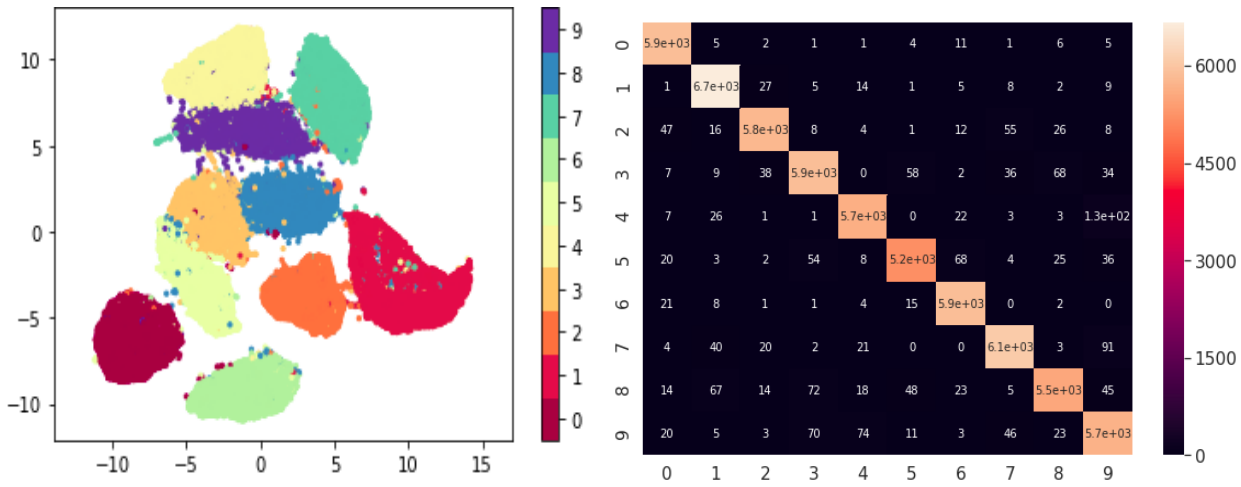
2. **Fashion-MNIST**
   The Fashion-MNIST is a dataset consisting of 60,000 training & 10,000 test examples published by *Zalando Research*. Each of the image in the dataset is a 28x28 grayscale image with associated label from 10 classes.
   Example:

3. **COIL-20**
   Columbia **O**bject **I**mage **L**ibrary is a database of 1440 images, which contains grayscale images of 20 objects, rotated in 360 degrees presenting different poses.
   Example:

Data Analysis

We are interested in analyzing the dimensionality reduction techniques on the selected datasets (MNIST, Fashion-MNIST, COIL-20). We will see that on one hand the reduction technique helps in visualizing the clusters in 2D and at the same time they also help in providing a great training in testing the accuracy in simple classification algorithms like ***KNeighborsClassifier***.

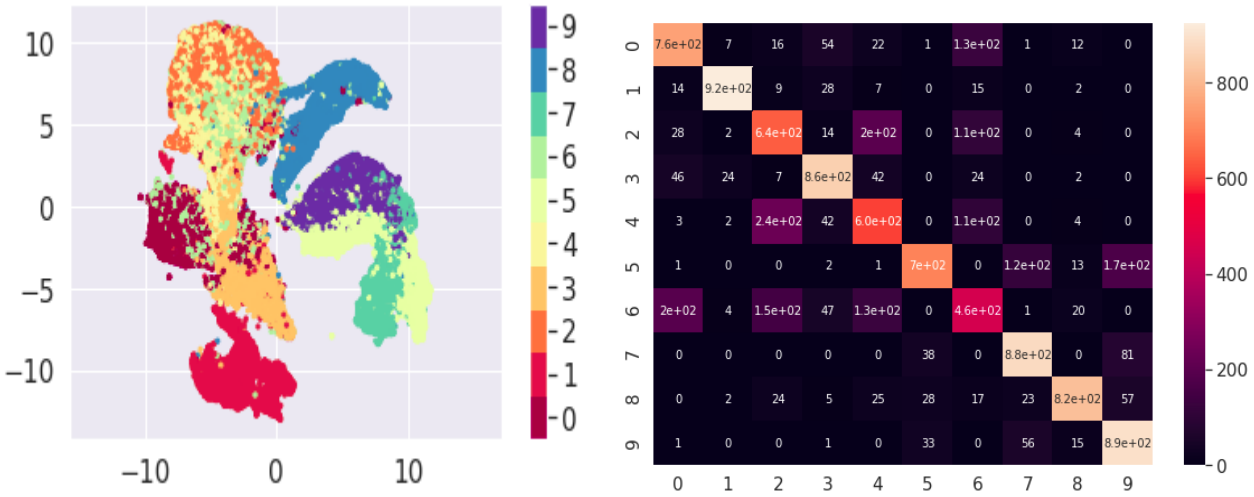
1. **UMAP**
   A. [MNIST Dataset]: Accuracy=0.8591



(a) (b)

(a) 2D representation after using UMAP. (b) Classification accuracy matrix

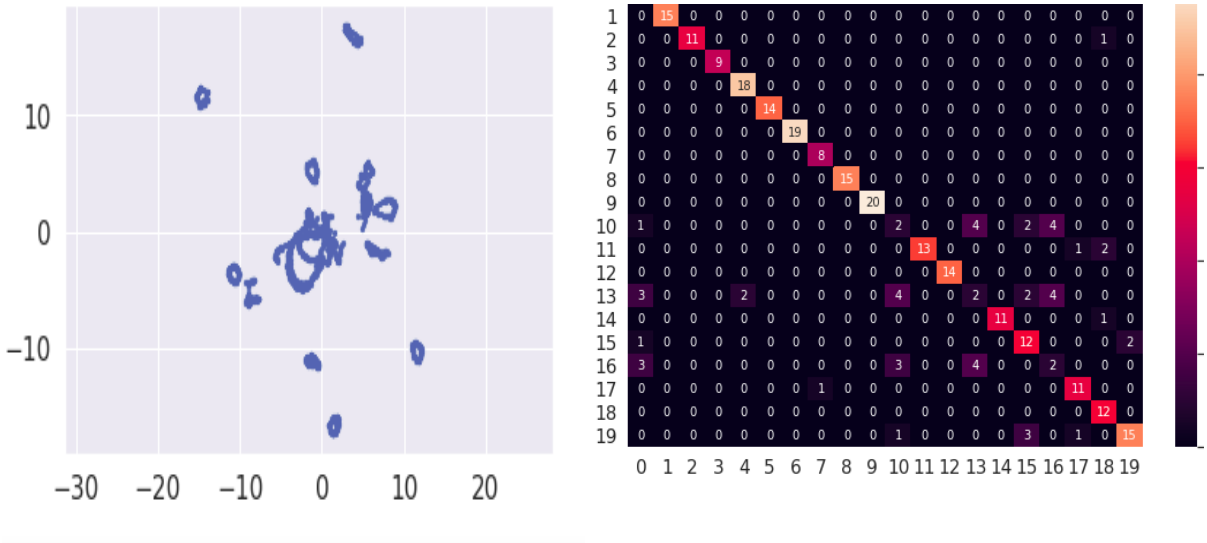
B. [Fashion - MNIST Dataset]: [Accuracy: 0.7532]



(a) (b)

(a) 2D representation after using UMAP. (b) Classification accuracy matrix

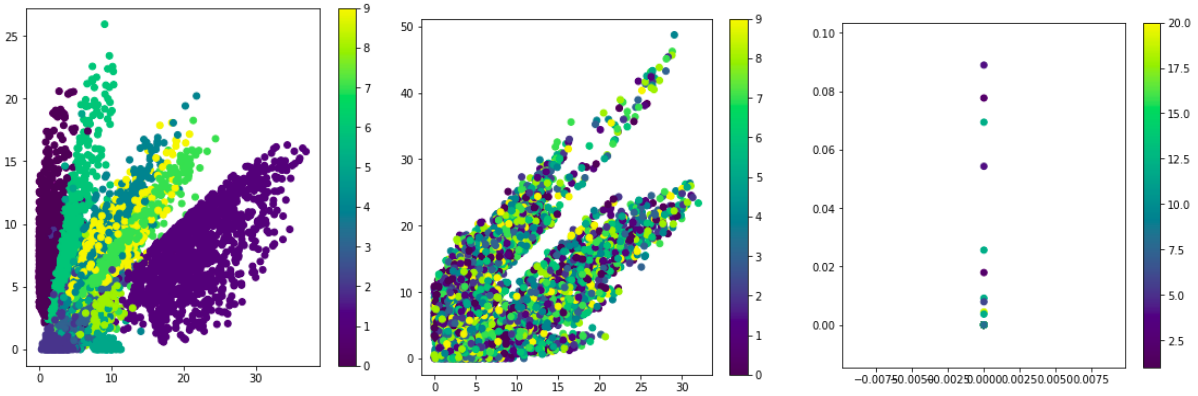
C. [COIL-20 Dataset]: [Accuracy: 0.8194444]



(a) **(b)**

(a) 2D representation after using UMAP. (b) Classification accuracy matrix

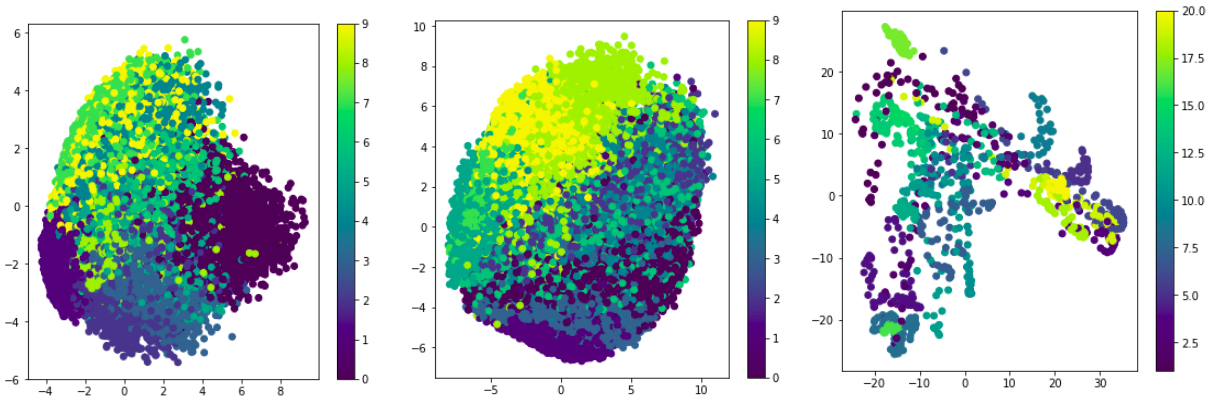
2. Auto Encoder*:



(a) **(b)** **(c)**

2D representation after using Autoencoder (a) MNIST (b) Fashion MNIST (c) COIL

3. PCA*



(a) **(b)** **(c)**

2D representation after using PCA (a) MNIST (b) Fashion MNIST (c) COIL-20

*Note:
We have skipped classification accuracy in case of Autoencoder (due to almost perfect classification accuracy) and PCA (due to very low classification accuracy)

Challenges

Initial exploration of dimensionality reduction techniques on these datasets reveal that selective algorithms works better on some dataset where fails to preserve some of latent information which in-turn reduces the classification accuracy.

Umap works amazingly fast and helps to represent the reduced dimension in 2D better than any explored algorithms. But it still faces issues in identifying complex shapes and similar objects. For example, Since Fashion-Mnist dataset consists of normalized images in grayscale have correct labels but results show that it fails in identifying ***class-6*** correctly.

Auto-encoder have been proved to be state-of-the-art techniques in identifying complex shapes, but have issues like computational challenges and not straightforward representation of latent dimensions. Since, our overall goal was to use auto-encoders as dimensionality reduction tools, which we believe may be worth exploring. On the flip side, auto encoders provide ~90% accuracy in classification in labelled data.

t-SNE & **MDS** were proven to work on non-linear as well as linear datasets, but they take huge time in processing. We found that it is taking more than 1.30 hours in processing MNIST dataset with standard computing resources.

PCA technique was processing fast into two chosen principal components but due to complex data shapes, it did not work well on any of the chosen dataset except COIL-20.

Future Scope

As of today, we have done the preliminary data analysis of the given datasets using three dimensionality reduction techniques (**UMAP**, **Autoencoder**, **PCA**). Furthermore, we plan to do an in-depth analysis of the given datasets to understand the data, carry out an exhaustive study for the above-mentioned techniques to better understand the problem statement and the techniques. We plan to apply these techniques on more datasets to compare the results.

Tasks Identified

1. Create a data pipeline of interesting high dimensional dataset.
2. Produce analysis of each dataset with labelled as well unlabeled setting. We believe that state-of-the-art deep neural networks like inception, imagenet, vggnet can help in providing labels for real life datasets.
3. Detailed analysis of ***Umap*** vs ***t-SNE***; since these two methods are most talked about in visualizing the datasets in reduced dimension; we want to evaluate this comparative study for minimum 5 more datasets.
4. Qualitative and Quantitative analysis of reduction algorithms and formation of extensive reasoning.

References

1. https://www.cs.toronto.edu/~hinton/science.pdf
2. https://blog.keras.io/building-autoencoders-in-keras.html
3. https://lvdmaaten.github.io/publications/papers/TR_Dimensionality_Reduction_Review_2009.pdf
4. http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
5. https://umap-learn.readthedocs.io/en/latest/benchmarking.html
6. https://www.sciencedirect.com/science/article/pii/S0020025514001741
7. https://www.nature.com/articles/nbt.4314
8. https://medium.com/@dan.allison/dimensionality-reduction-with-umap-b081837354dd
9. https://nbisweden.github.io/excelerate-scRNAseq/session-dim-reduction/lecture_dimensionality_reduction.pdf
10. https://towardsdatascience.com/autoencoders-in-keras-c1f57b9a2fd7
11. https://medium.com/datadriveninvestor/deep-autoencoder-using-keras-b77cd3e8be95
12. https://github.com/zalandoresearch/fashion-mnist