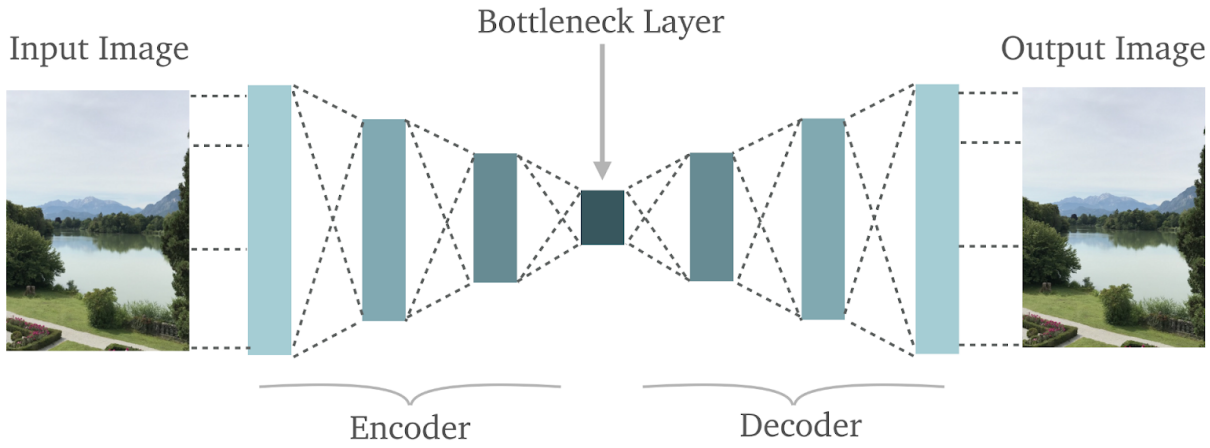
DIMENSIONALITY REDUCTION WITH AUTOENCODERS

Mainak Ghosh
Roll : 91/MCA/170003
Registration No. : 133-1111-0283-17
Project Supervisor :
Mr. Dibyendu B. Seal

TABLE OF CONTENTS

INTRODUCTION- WHAT IS AUTOENCODER ?

An autoencoder is a type of artificial neural network that learns how to efficiently compress the data then learns how to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible.



INTRODUCTION- GOALS OF AUTOENCODERS

- Dimensionality Reduction and Feature extraction.
- Anomaly Detection
- Image Denoising
- Generative Models

PROBLEM STATEMENT

The objective of this project is threefold.

- Our first objective is to build different types of autoencoders.

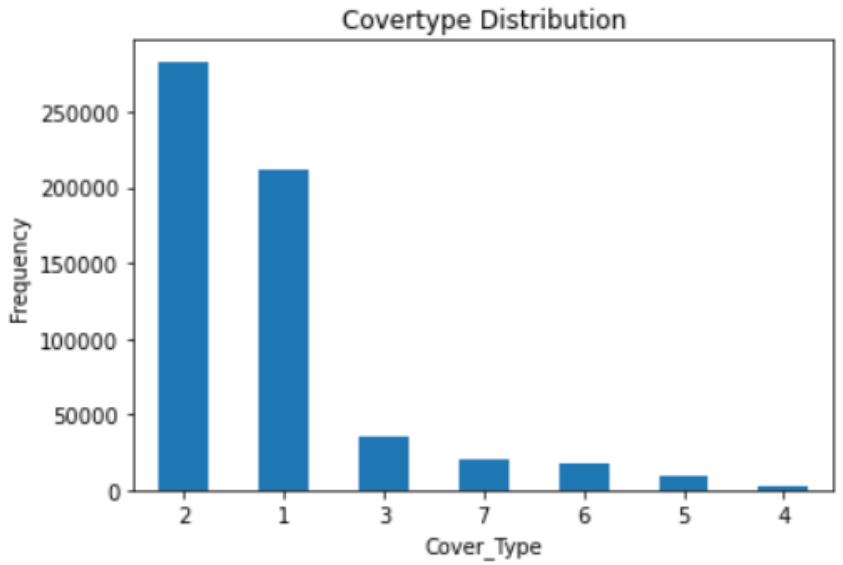
- Our second objective is to compare the dimensionality reduction ability of those different kinds of autoencoders on different datasets.

- Our third objective is to compare the results with other state-of-the-art dimensionality reduction techniques.

DATA ACQUISITION

No	Name	Instances	Features	Source
1.	Forest cover type dataset	581012	54	https://www.kaggle.com/uciml/forest-cover-type-dataset
2.	MNIST handwritten digits database	70000	784	https://www.kaggle.com/oddrationale/mnist-in-csv
3.	Breast Cancer Wisconsin (Diagnostic) Dataset	569	30	https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

DATA PREPARATION - FOREST COVER TYPE DATA

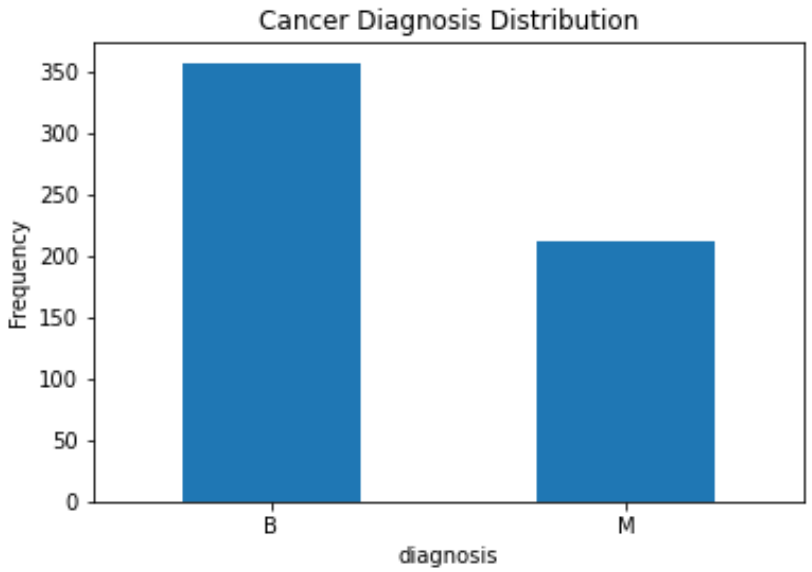


- The distribution of cover types is highly imbalanced. So we used an under sampling method with the help of imbalanced-learn API. After under sampling we have 19229 instances to work on.
- To train the autoencoders we dropped the target label “cover_type” from the train dataset and stored it separately to use later on in classification.
- The range of values of different features in the data vary widely, so we used min-max scaling to normalize the data between 0 and 1.

DATA PREPARATION - MNIST DIGIT DATASET

- We have dropped the “values” column to only retain the columns with the pixel values.
- After that those DataFrames are converted into NumPy arrays. Each array contains 784 columns.
- Since the pixel values varied from 0 to 255, to normalize the arrays between 0 and 1, both the arrays are divided by 255. We showed the first 10 sample data after normalization.

DATA PREPARATION - BREAST CANCER DATASET



- The distribution of diagnosis is fairly balanced.
- To train the autoencoders we dropped the target label “diagnosis” from the train dataset and stored it separately to use later on in classification.
- To normalize the data between 0 and 1 we used min-max scaling.

DATA ANALYSIS - SIMPLE AUTOENCODER

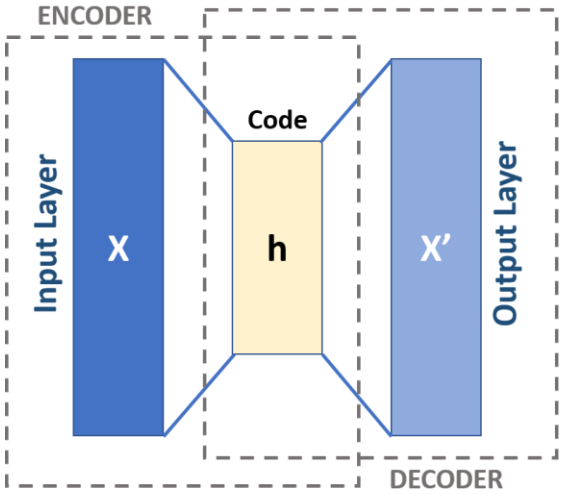
The encoded output is: $h = f_e(x) = s_e(W_e x + b_e)$

The reconstruction of original input is: $x_r = f_d(h) = s_d(W_d h + b_d)$.

Where $f_d : R^d \rightarrow R^h$ and $f_e : R^h \rightarrow R^d$ are encoding and decoding functions respectively. W_e and W_d are the weights of the encoding and decoding layers. b_e and b_d are the biases for the two layers. s_e and s_d are elementwise nonlinear functions in general, and common choices are sigmoidal functions.

For training, we want to find a set of parameters $\Theta = \{W_e, W_d, b_e, b_d\}$ that minimize the reconstruction error :

$$\sum_{x \in D} L(x, x_r)$$



DATA ANALYSIS - SIMPLE AUTOENCODER

For the forest cover type data:

- Model Structure : <54,3,54>.
- Activation function: RELU is used in encoder and Sigmoid is used in decoder.
- Loss function : MSE
- Optimizer : Adam

For the MNIST data:

- Model Structure : <784,32,784>.
- Activation function: RELU is used in encoder and Sigmoid is used in decoder.
- Loss function : Binary cross-entropy
- Optimizer : Adadelta

For the Breast cancer data:

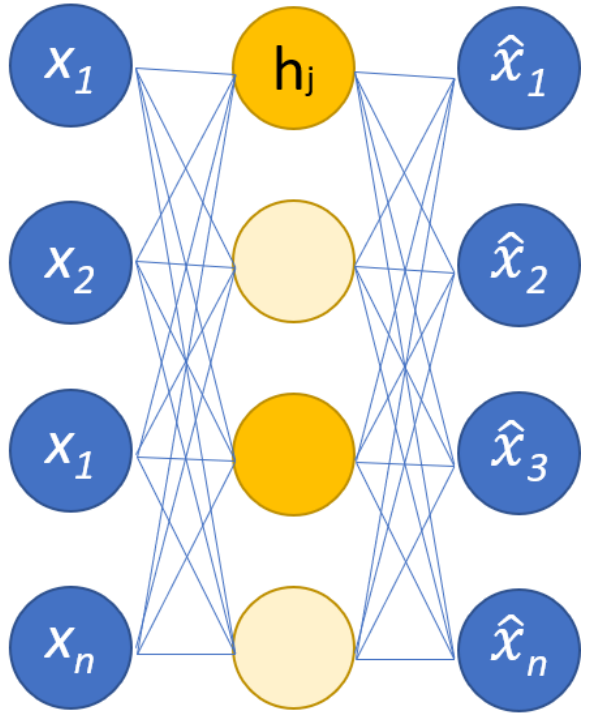
- Model Structure : <30,3,30>.
- Activation function: RELU is used in encoder and Sigmoid is used in decoder.
- Loss function : MSE
- Optimizer : Adam

DATA ANALYSIS - SPARSE AUTOENCODER

In a sparse autoencoder, a sparsity penalty $\Omega(h)$ is added on the code layer h.

This involves constructing a loss function $L(x, x_r) + \Omega(h)$, where one term encourages our model to be sensitive to the inputs (ie. reconstruction loss $L(x, x_r)$ and an added regularize/sparsity penalty $\Omega(h)$ discourages memorization /overfitting.

The penalty $\Omega(h)$ encourages the model to activate (i.e. output value close to 1) some specific areas of the network on the basis of the input data, while forcing all other neurons to be inactive (i.e. to have an output value close to 0).



- To achieve sparsity, we added L1 regularization of 1e-6 in the encoding layer of the simple autoencoder for each dataset.

DATA ANALYSIS - DEEP AUTOENCODER

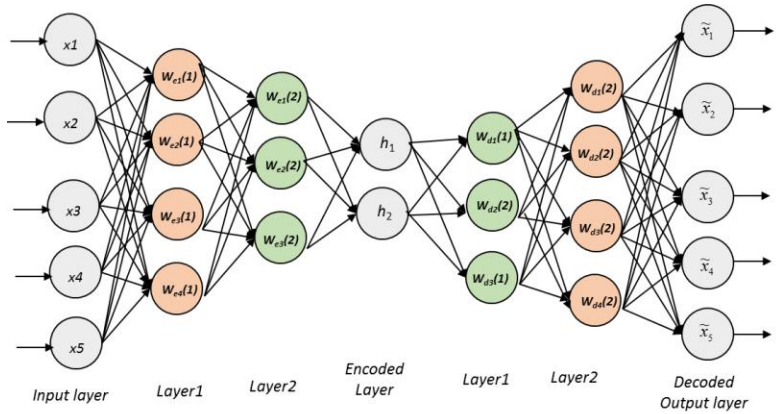
If an autoencoder has multiple hidden layers, they are called Deep autoencoders. In general, an N-layer deep autoencoder with parameters $\Theta = \{\Theta^i \mid i \in \{1, 2, ...N\}\}$, where $\Theta^i = \{W_e^i, W_d^i, b_e^i, b_d^i\}$ can be formulated as follows:

$$h^i = f_e^i(h^{i-1}) = s_e^i(W_e^i h^{i-1} + b_e^i)$$

$$h_r^i = f_d^i(h_r^{i+1}) = s_d^i(W_d^i h_r^{i+1} + b_d^i)$$

$$h^0 = x$$

The deep autoencoder architecture therefore contains multiple encoding and decoding stages made up of a sequence of encoding layers followed by a sequence of decoding layers. Therefore the deep autoencoder has a total of 2N layers.



DATA ANALYSIS – DEEP AUTOENCODER

For the forest cover type data:

- Model Structure : <54,27,9,3,9,27,54>.
- Activation function: RELU but in last layer of the decoder we used sigmoid.
- Loss function : MSE
- Optimizer : Adam

For the MNIST data:

- Model Structure : <784,128,64,32,64,128,784>.
- Activation function: RELU but in last layer of the decoder we used sigmoid.
- Loss function : Binary cross-entropy
- Optimizer : Adadelta

For the Breast cancer data:

- Model Structure : <30,20,10,3,10,20,30>.
- Activation function: RELU but in last layer of the decoder we used sigmoid.
- Loss function : MSE
- Optimizer : Adam

DATA ANALYSIS - DENOISING AUTOENCODER

The idea of denoising autoencoders is to take a partially corrupted input and train the autoencoder to recover the original undistorted input. In this way, the model is forced to learn representations that are useful.

The training process of a denoising autoencoder works as follows:

1. The initial input x is corrupted into x_c via some corruption process. Formally, $x_c \sim q(x_c \mid x)$, where $q(\cdot \mid x)$ is some corruption process over the input x.
2. The corrupted input x_c then mapped to a hidden representation with the same process of the standard autoencoder,

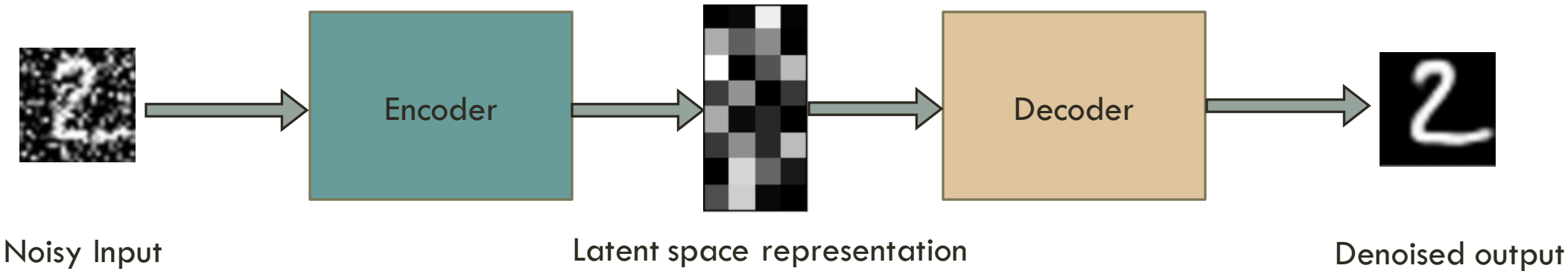
$$h = f_e(x) = s_e(W_e x_c + b_e)$$

3. From the hidden representation the model reconstructs x_r following the same process as the standard autoencoder.

Denoising the inputs intentionally are particularly helpful in case of an overfitting problem of a model. Denoising autoencoder also used for data denoising and image denoising.

DATA ANALYSIS - DENOISING AUTOENCODER

- To create noise we create an array of random numbers with mean of 0 and the variance of 1 and with the shape of the training data. Then we multiplied the array with some noise factor. Then we add the noise to the original data to create the noisy train data.
- In the training process, we train the model with noisy data as input and clean data as target output allowing the model to learn essential features of the given data.

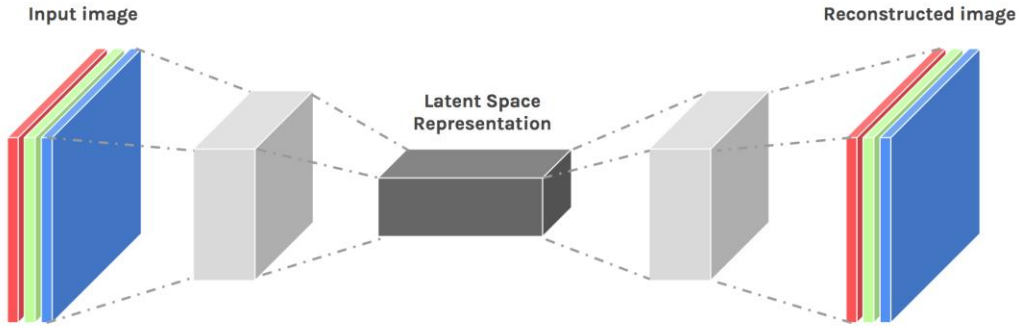


DATA ANALYSIS - CONVOLUTIONAL AUTOENCODER

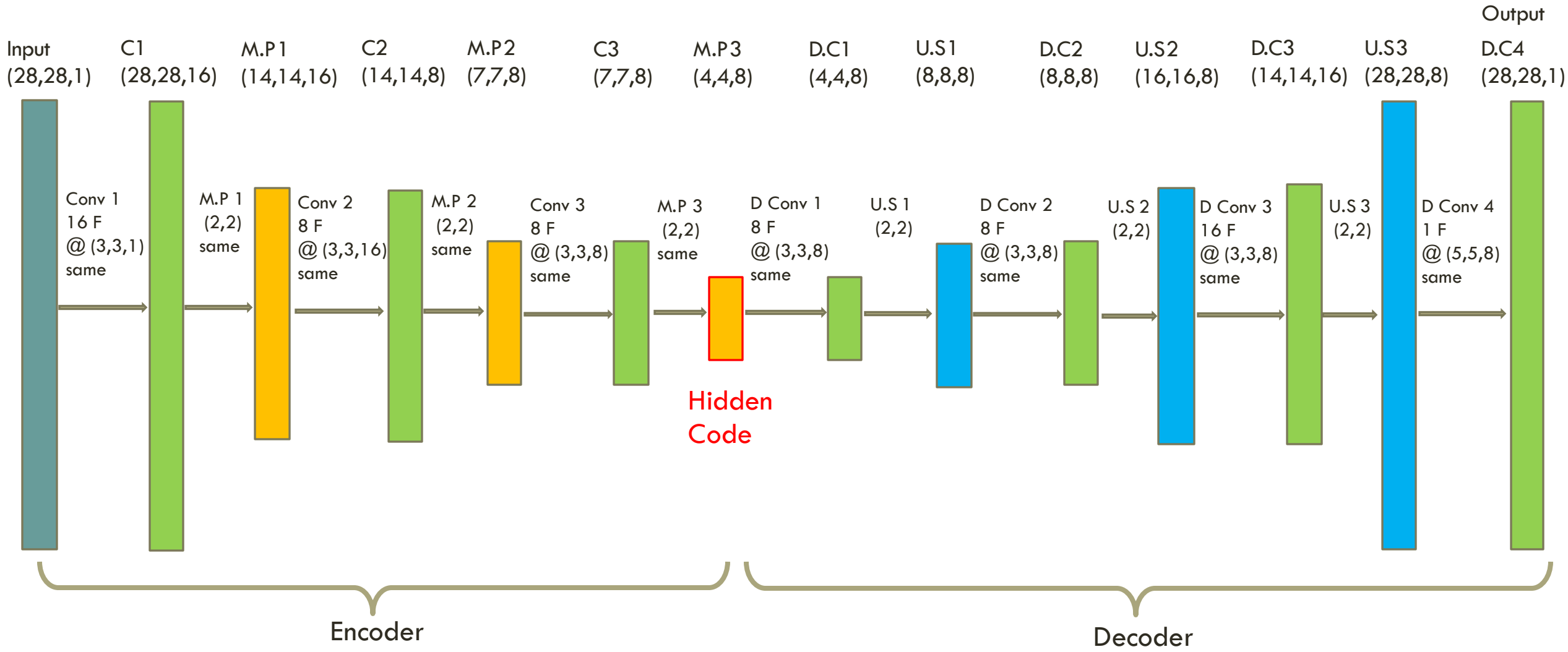
- The Convolutional Autoencoder is extended from autoencoder by instantiating encoder function and decoder function with convolutional neural networks (CNN).
- The basic building blocks of CNN are convolutional layers and pooling layers, a convolutional layer consists of multiple convolutional nodes whose inputs are 2-dimensional feature maps, the learning parameters are the elements of filter matrices.
- A pooling layer is also called a sampling layer, the nodes of a pooling layer are obtained by sampling the corresponding nodes of a convolutional layer.
- Consequently, if the number of nodes of the convolutional layer is m, then the number of nodes of the pooling layer is also m. Because a CNN is a feedforward neural network, the CAE can be trained with gradient descent algorithm or stochastic gradient descent algorithm.

DATA ANALYSIS - CONVOLUTIONAL AUTOENCODER

- So to use convolutional neural network we reshaped the dimension of the data accordingly.
- For the encoder part we used convolutional layer 1d(2d) and max pooling 1d(2d) layer. Also Batch normalization layer to normalize the data for better results. For each convolutional layer we used Relu as the activation function.
- Similarly in the decoder part we used convolutional layer 1d(2d) and up sampling 1d(2d) layer. For each convolutional layer there is an up sampling 1d(2d) layer to upscale the dimension of the tensor.



CONVOLUTIONAL AE MODEL STRUCTURE FOR MNIST DATA



DATA ANALYSIS - STACKED AUTOENCODER

The structure of SAEs is stacking autoencoders into hidden layers by an unsupervised layer-wise learning algorithm. So the SAEs based method can be divided into three steps:

- Train the first autoencoder by input data and obtain the learned feature vector.
- The feature vector of the former layer is used as the input for the next layer, and this procedure is repeated until the training completes.
- After all the hidden layers are trained, a backpropagation algorithm is used to minimize the cost function and update the weights to achieve fine-tuning.

DATA ANALYSIS - STACKED AUTOENCODER

For the forest cover type data:

- AE 1 Model Structure : <54,27,54>,
- AE 2 Model Structure : <27,9,27>,
- AE 3 Model Structure :<9,3,9>,
- Deep AE Model Structure :<54,27,9,3,9,27,54>.
- Activation function: RELU in encoder and used sigmoid in the output layer of the decoder.
- Loss function : MSE
- Optimizer : Adam

For the Breast cancer data:

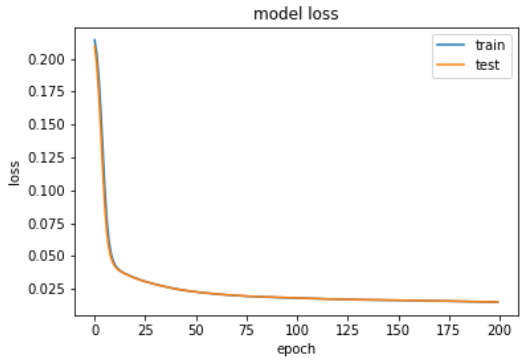
- AE 1 Model Structure : <30,20,30>,
- AE 2 Model Structure : <20,10,20>,
- AE 3 Model Structure : <10,3,10>,
- Deep AE Model Structure : <30,20,10,3,10,20,30>.
- Activation function: RELU in Encoder and used sigmoid in the output layer of the decoder.
- Loss function : MSE
- Optimizer : Adam

For the MNIST data:

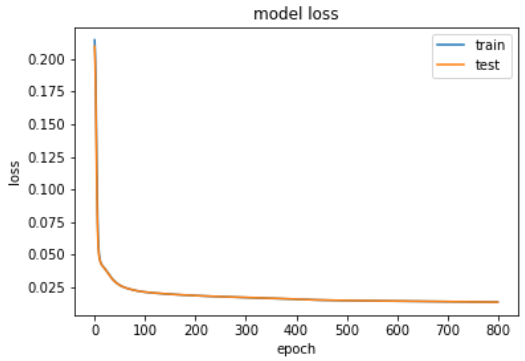
- AE 1 Model Structure : <784,128,784>,
- AE 2 Model Structure : <128,64,128>,
- AE 3 Model Structure :<64,32,64>,
- Deep AE Model Structure :<784,128,64,32,64,128,784>.
- Activation function: RELU in Encoder and Sigmoid in Decoder
- Loss function : Binary cross-entropy
- Optimizer : Adadelta

RESULTS AND DISCUSSION – FOREST COVER DATA

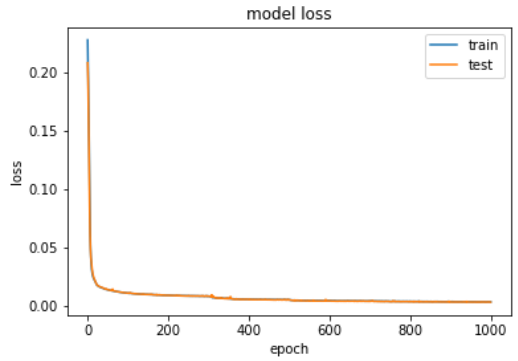
- Comparison between model loss of the Forest cover data.



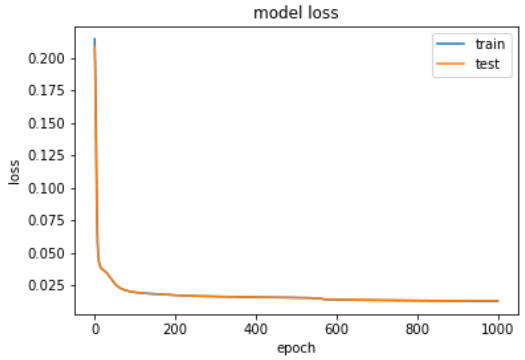
Simple AE



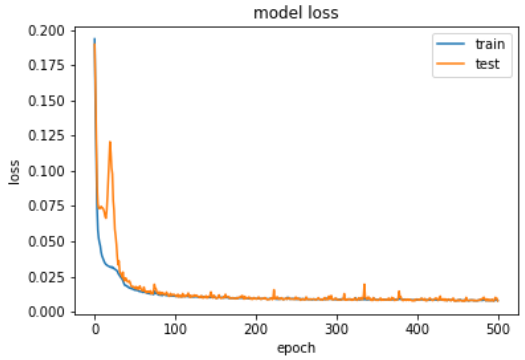
Sparse AE



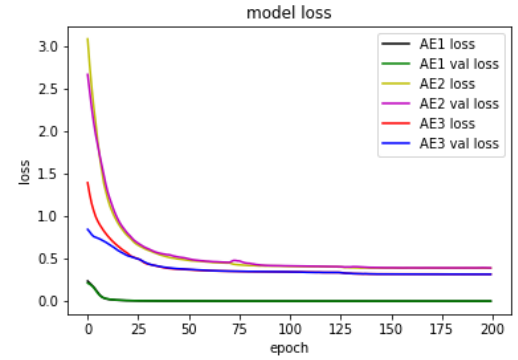
Deep AE



Denoising AE



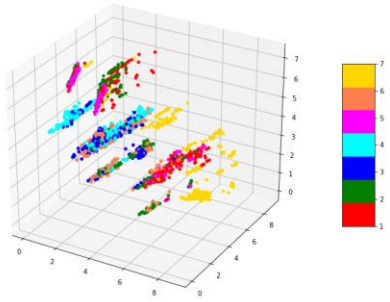
Convolutional AE



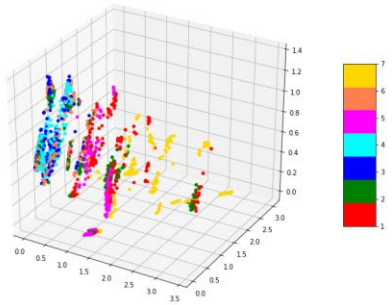
Stacked AE

RESULTS AND DISCUSSION – FOREST COVER DATA

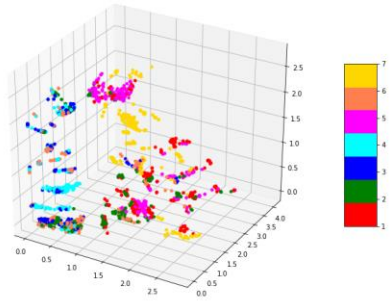
- Data Visualization in three dimensional space using the encoded representation.



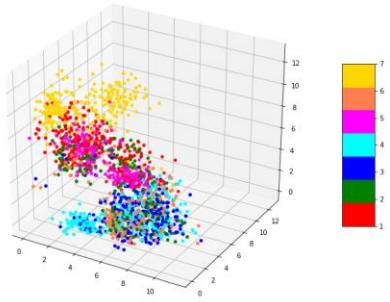
Simple AE



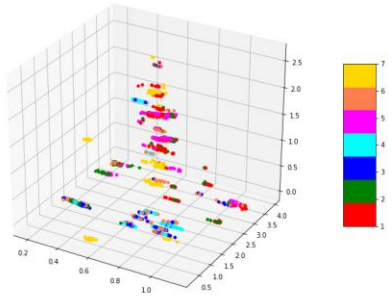
Sparse AE



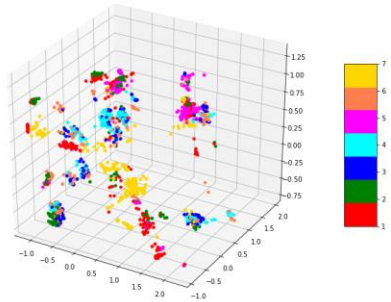
Deep AE



Denoising AE



Convolutional AE



Stacked AE

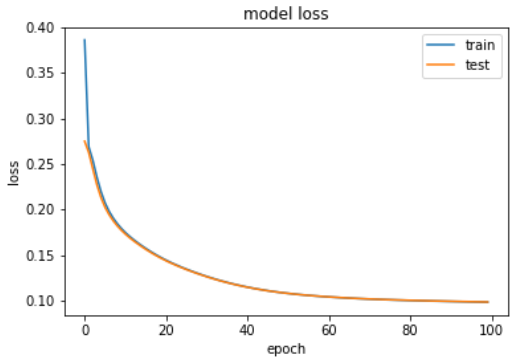
RESULTS AND DISCUSSION – FOREST COVER DATA

- Comparison between model loss and KNN score of the Forest cover data

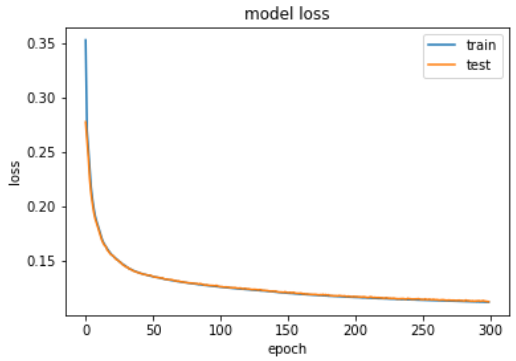
	No. of features (Original)	No. of features (Reduced)	Training Loss	Validation Loss	Test Loss	KNN Score
Simple AE	54	3	0.150	0.0150	0.0147	0.7779
Sparse AE	54	3	0.0133	0.0133	0.0120	0.7847
Deep AE	54	3	0.0030	0.0028	0.0028	0.8013
Convolutional AE	54	3	0.0079	0.0078	0.0077	0.7108
Denoising AE	54	3	0.0131	0.0134	0.0130	0.5605
Stacked AE	54	3	8.0239e-04 0.1104 0.8801	0.0010 0.1131 0.9174	0.02917	0.8169
PCA	54	3	-	-	-	0.8263

RESULTS AND DISCUSSION – MNIST DATA

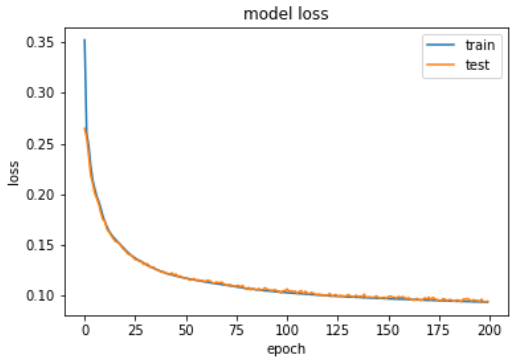
- Comparison between model loss of the MNIST handwritten digits dataset.



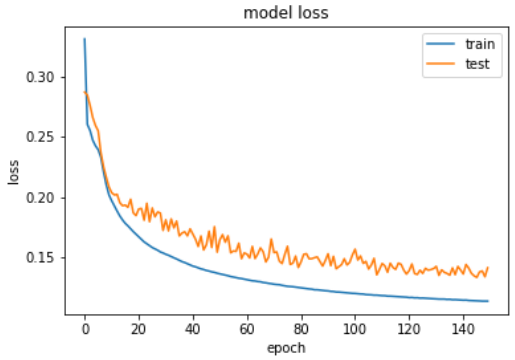
Simple AE



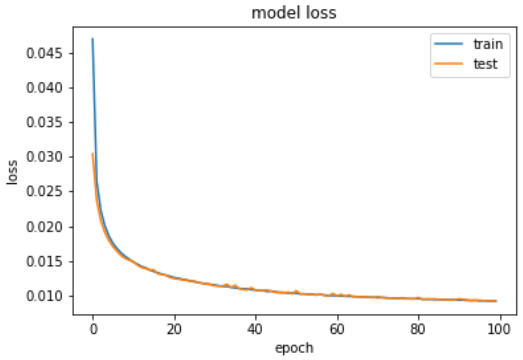
Sparse AE



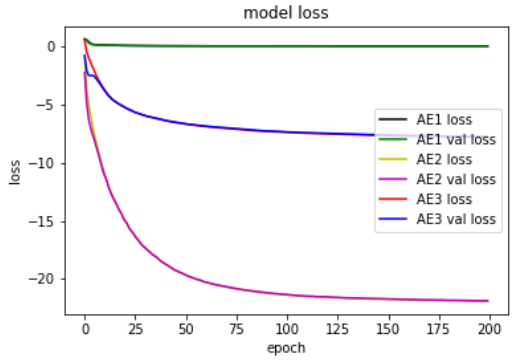
Deep AE



Denoising AE



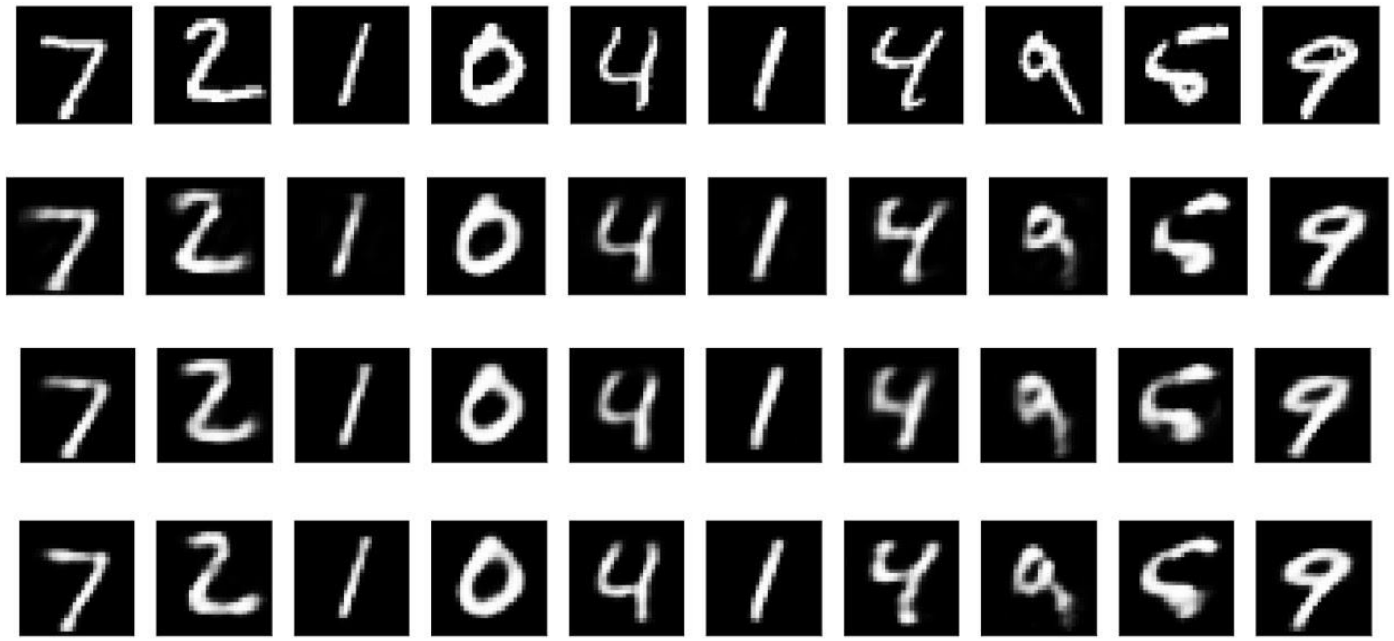
Convolutional AE



Stacked AE

RESULTS AND DISCUSSION – MNIST DATA

- Comparison between reconstructed output with the original input



Original input

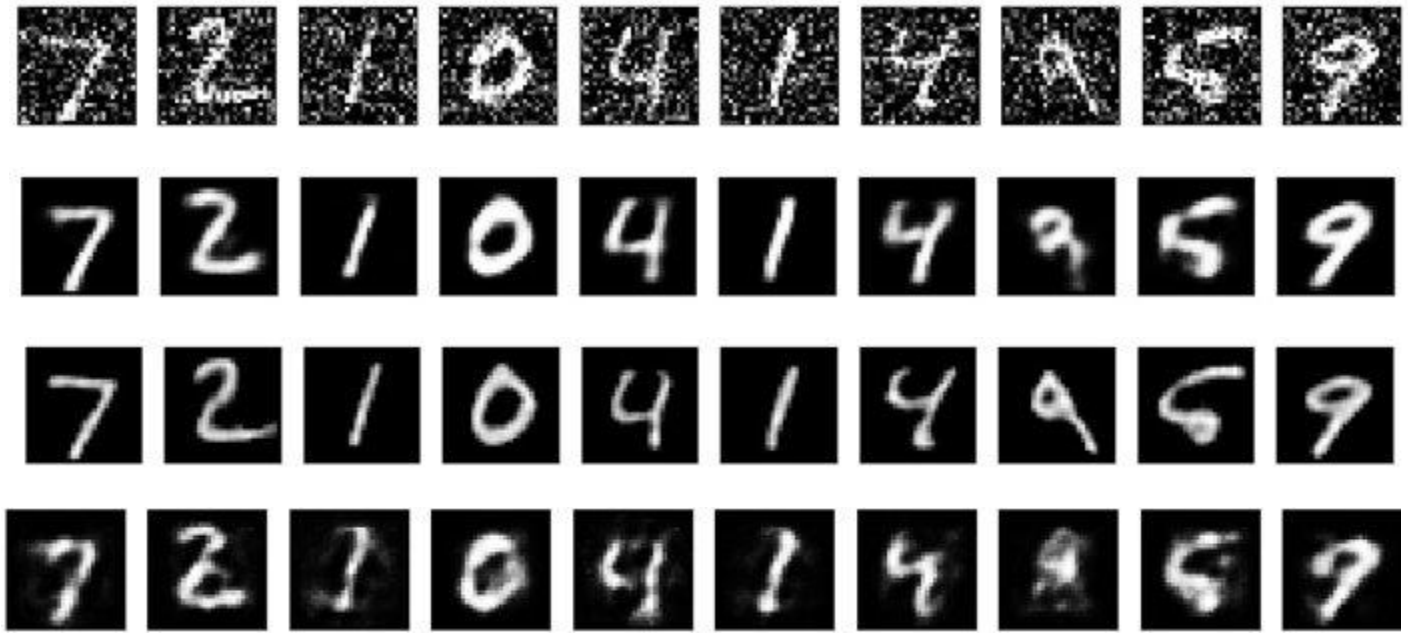
Simple AE output

Sparse AE output

Deep AE output

RESULTS AND DISCUSSION – MNIST DATA

- Comparison between reconstructed output with the original input



Noisy input

Denoising AE output

Convolutional AE output

Sparse AE output

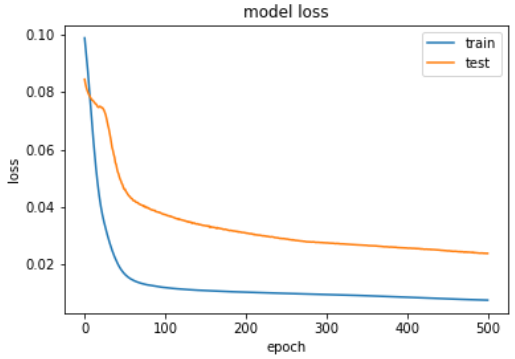
RESULTS AND DISCUSSION – MNIST DATA

- Comparison between model loss and KNN score of the MNIST data.

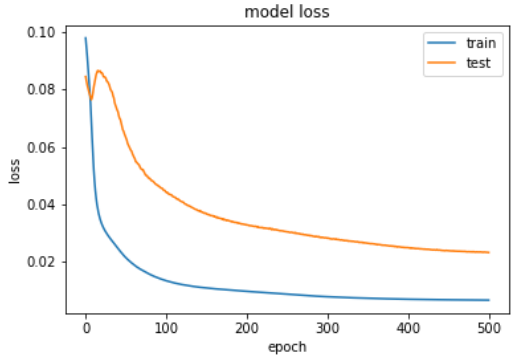
	No. of Features (Original)	No. of Features (Reduced)	Training Loss	Validation Loss	Test Loss	KNN Score
Simple AE	784	32	0.0984	0.0985	0.0968	0.9678
Sparse AE	784	32	0.1126	0.1131	0.1066	0.9615
Deep AE	784	32	0.0939	0.0948	0.0934	0.9725
Convolutional AE	784	128	0.0092	0.0092	0.0089	0.952
Denoising AE	784	32	0.1137	0.1413	0.1169	0.9658
Stacked AE	784	32	0.0702 -21.9300 -7.7876	0.0708 -21.9021 -7.7876	0.1514	0.9363
PCA	784	32	-	-	-	0.9723

RESULTS AND DISCUSSION – BREAST CANCER DATA

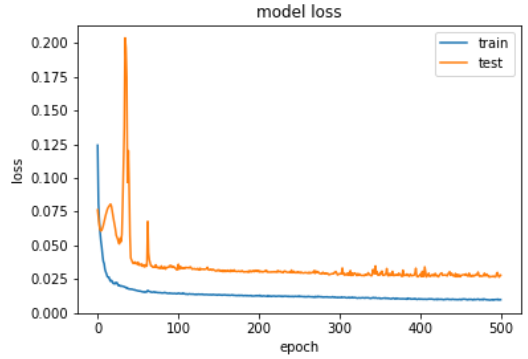
- Comparison between model loss of the Breast cancer data.



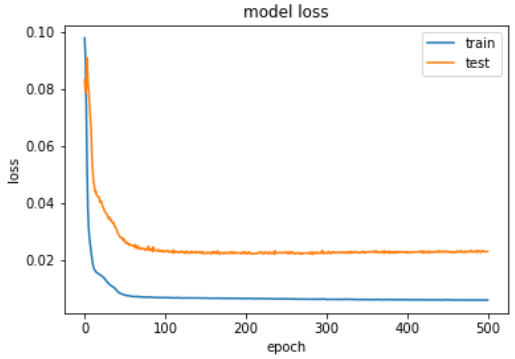
Simple AE



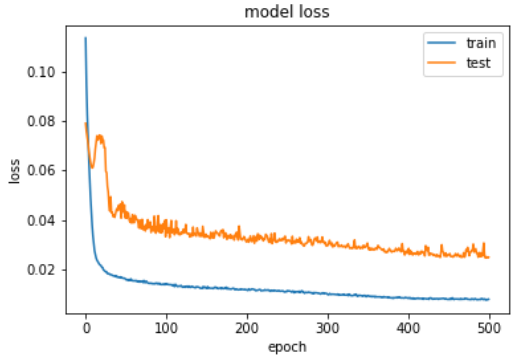
Sparse AE



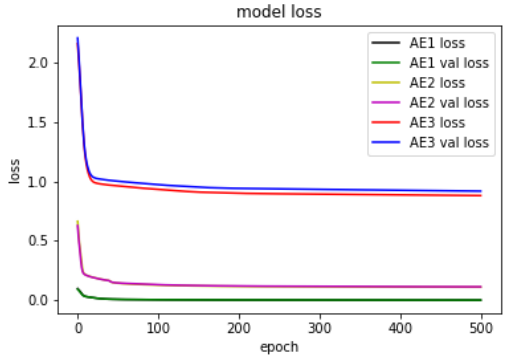
Deep AE



Denoising AE



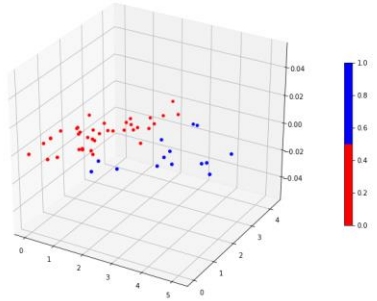
Convolutional AE



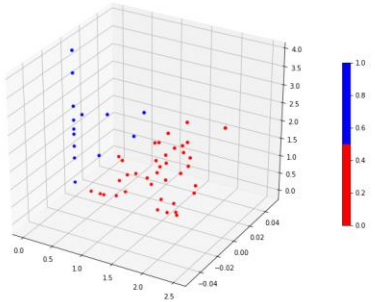
Stacked AE

RESULTS AND DISCUSSION – BREAST CANCER DATA

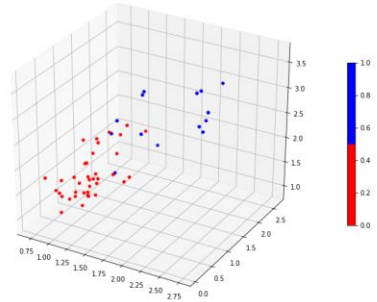
- Data Visualization in three dimensional space using the encoded representation.



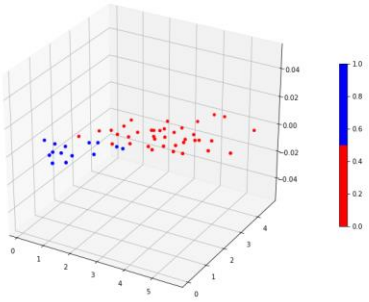
Simple AE



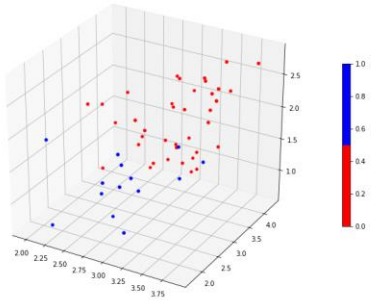
Sparse AE



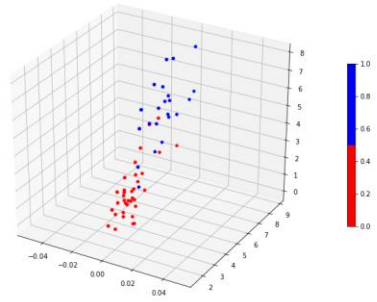
Deep AE



Denoising AE



Convolutional AE



Stacked AE

RESULTS AND DISCUSSION – BREAST CANCER DATA

- Comparison between model loss and KNN score of the Breast cancer data.

	No. of Features (Original)	No. of Features (Reduced)	Training Loss	Validation Loss	Test Loss	KNN Score
Simple AE	30	3	0.0075	0.0238	0.0289	0.7115
Sparse AE	30	3	0.0068	0.0233	0.0287	0.8846
Deep AE	30	3	0.0053	0.0209	0.0258	0.8846
Convolutional AE	30	3	0.0078	0.0249	0.0261	0.8461
Denoising AE	30	3	0.0058	0.0229	0.0295	0.8653
Stacked AE	30	3	8.0239e-04 0.1104 0.8801	0.0010 0.1131 0.9174	0.0812	0.8596
PCA	30	3	-	-	-	0.9807

CONCLUSION

- From the model loss and classification results, we could see that using Deep AE and Stack AE performs better in feature reduction than other models for forest cover type data.
- Convolutional AE performs better in feature reduction than other models for MNIST data.
- Deep AE and Sparse AE performs better in feature reduction than other models for breast cancer diagnostic data.
- From the results compared with the PCA transformation, we can conclude that for the Breast cancer data PCA transformation works far better than autoencoders due to the fact that there are a small number of instances in the dataset. For the other two datasets the results are almost the same in best case scenario.

FUTURE WORK

- Finding out the minimum number of feature dimensions for each dataset so that the best possible classification accuracy can be achieved or mean squared error loss is the lowest.
- Performing other state-of-the-art dimension reduction techniques such as T-SNE, UMAP, PHATE to compare the result.
- We know that the weight initialization of the network can give different results. Therefore, discovering how to pre-train the initial weight effectively so that the performance of the models increases.
- Here I have applied only one classification method but in future more methods can be applied.

REFERENCES

[1] T. Wen and Z. Zhang, "Deep Convolutional Neural Network and Autoencoders-Based Unsupervised Feature Learning of EEG Signals," in IEEE Access, vol. 6, pp. 25399-25410, 2018, doi: 10.1109/ACCESS.2018.2833746.

[2] Wang, Y., Yao, H., Zhao, S., & Zheng, Y. (2015). Dimensionality reduction strategy based on auto-encoder. ICIMCS '15.

[3] Sakurada, Mayu and Takehisa Yairi. "Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction." MLS DA14 (2014).

[4] Majeed, Sajid & Mansoor, Yusra & Qabil, Sana & Majeed, Farooq & Khan, Behraj. (2020). Comparative analysis of the denoising effect of unstructured vs. convolutional autoencoders. 1-5. 10.1109/ICETST49965.2020.9080731.

[5] Lingheng Meng, Shifei Ding1, Yu Xue. "Research on denoising sparse autoencoder". DOI: 10.1007/s13042-016-0550-y.

[6] J. Zhai, S. Zhang, J. Chen and Q. He, "Autoencoder and Its Various Variants," 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan, 2018, pp. 415-419, doi: 10.1109/SMC.2018.00080.

REFERENCES

[7]Yingbo Zhou, Devansh Arpit, Ifeoma Nwogu, Venu Govindaraju: "Is Joint Training Better for Deep Auto-Encoders?" in arXiv:1405.1380v4 [stat.ML] 15 Jun 2015

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

[9] Guifang Liu, Huaiqian Bao, and Baokun Han. (2018), A Stacked Autoencoder-Based Deep Neural Network for Achieving Gearbox Fault Diagnosis, https://doi.org/10.1155/2018/5105709

[10] Shivappriya, Sn & Rajaguru, Harikumar. (2019). Performance Analysis of Deep Neural Network and Stacked Autoencoder for Image Classification: Intelligence and Sustainable Computing. 10.1007/978-3-030-02674-5_1.

[11] Almotiri, Jasem & Elleithy, Khaled & Elleithy, Abdelrahman. (2017). Comparison of autoencoder and Principal Component Analysis followed by neural network for e-learning using handwritten recognition. 1-5. 10.1109/LISAT.2017.8001963.

[12] Meng, Qinxue, Daniel Catchpoole, David Skillicom and Paul J. Kennedy. "Relational autoencoder for feature extraction." 2017 International Joint Conference on Neural Networks (IJCNN) (2017): 364-371.

THANK YOU!