Spectral Clustering with Ensemble of Autoencoders

Prithul Sarker
University of Nevada, Reno
Department of Computer Science and Engineering
prithulsarker@nevada.unr.edu

Abstract

Spectral clustering is one of the most popular clustering algorithms of modern times. When using spectral clustering on unaltered original input data, because of the high dimensions of the input, it shows difficulty in approximating proper Laplacian matrix. The same result is shown in evaluation section. To overcome this issue, reducing the dimension is a feasible solution. In this paper, I propose ensemble of autoencoders to use spectral clustering method on the encoded image. I use multiple convolutional autoencoders to reduce the dimensions of the images. Based on the outputs from ensemble of autoencoders, the best model is chosen. The encoded images from the models are used as the input of spectral clustering method. Experiments on MNIST dataset empirically validate the usage of ensemble of autoencoders.

1. Introduction

Deep learning has evolved together with the explosion of data in all forms and from every sector of our life. This huge amount of data which is also called 'big data' is drawn from many types of platforms which include social media, medical science, customer behavior and others. In most cases, the data is normally unstructured, and it could take a lot of time for humans to label those data and make them structured. Unstructured data makes up almost 80% or more of the enterprise data, and the percentage is growing every day.[1] In recent times, scientists all around the world is working hard to learn and glean from this massive amount of unstructured data. Because of the noticeable development in the field of deep learning with big data, it is no wonder that everyone is facing towards deep learning and its deep neural networks.[2] Based on the labelling of the data we have, deep learning is divided in two groups: supervised learning, and unsupervised learning.

Although deep neural networks do not have significant success in the unsupervised learning area, the progress is noteworthy. While supervised learning is much easier for deep learning programs to process, unsupervised learning presents a more challenging task. Also, it can be easily said that unsupervised learning is going to be the challenge for coming years as the amount of unstructured data is increasing day by day. The advantage of unsupervised deep leaning is numerous, and it has the potential to unlock previously unsolvable problems. Unsupervised learning algorithms derive insights directly from the data and work as summarizing the data and grouping it.[3] However, the available machine learning clustering algorithms for unsupervised data are somewhat ineligible to group the data. One of the reasons for that is the algorithms want to group them in higher dimension which eventually gets failed. In most clustering algorithms, the distance metric used is the Euclidean distance. Each distance measure gives different outcomes for that. As dimension increases, distance measures like Euclidean distance or Manhattan distance get into trouble. Therefore, most clustering algorithm is randomized, and each time these are run, different clustering is found even though same datasets with same dimension is used.

Spectral clustering is different than most other clustering algorithms and it is one of the most popular modern clustering algorithms. The reason behind its popularity is it is simple to implement, can be solved efficiently by standard linear algebra, and in most cases, spectral clustering outperforms other traditional algorithms such as k-means algorithm. This clustering algorithm uses eigenvalues and eigenvectors of Laplacian matrix derived from the dataset.[4]

To use deep learning methods in unsupervised learning, autoencoder is one of the most used networks. An autoencoder is an unsupervised machine learning algorithm that sets the target values equal to the input values and uses a bottleneck layer in the middle.[5] An autoencoder has two main parts: encoder and decoder. The encoder part of the autoencoder reduces the dimension of the input data and compresses the input into a latent space representation. The decoder just decodes the encoded image and reconstructs to its original input size. Based on the error of the original and reconstructed data, the loss function is determined and is trained on. The middle layer between the encoder and decoder is the bottleneck network.

In modern research studies, a successful approach has been to reduce the variance of neural network models. To mention this approach, rather than training a single model training multiple models and finally combining the prediction from these models has been fruitful. To combine multiple models and finally based on the metrics, choosing a final model is called ensemble models. When ensemble models are trained using deep learning method, the learning is called ensemble deep learning. It helps to avoid generalization error.

In this paper, an ensemble of auto-encoders and spectral clustering on the encoded data is proposed. It is hypothesized that leveraging a mixture of autoencoders, and clustering method may further improve performance on both supervised and unsupervised data and may play a vital role in big data.

2. Related Work

Ensemble deep learning, a machine learning technique, has attracted substantial attention from machine learning community and achieved great success in various artificial intelligence applications due to alleviating weights initialization and better generalization. Dietterich in AI magazine listed ensemble learning as the first of the four research directions in machine learning.[6] Since ensemble learning can improve the generalization ability of learning systems, researchers have been working on its theoretical algorithms and applications as early as 1990. The progress of ensemble learning has been remarkable in the last three decades.[7] A couple of recent individual and joint works of deep learning, ensemble learning, and bioinformatics and their achievements and shortcomings are discussed below.

Geddes et al. in "Autoencoder-based cluster ensembles for single-cell RNA-seq data analysis" (2019) paper, proposed an autoencoder-based ensemble clustering and evaluated the method's validity using different datasets of sc-RNA. They applied random subspace projection due to the higher dimension of the datasets and then applied encoder training. According to the metrics and the datasets used for evaluation, the ensemble of clusters improved performance about 30% on average and the cell type identification is more accurate when ensemble of autoencoder-based clustering framework is being used.[8] However, they failed to make this algorithm optimum across scRNA-seq datasets globally. An interesting insight of the paper is they confirmed using a comparison that autoencoder based cluster ensemble outperforms PCA-based clustering in almost all cases using Wilcoxon Rank Sum test.

In "A deep learning-based multi-model ensemble method for cancer prediction" (2018) paper by Xiao, Y. et al., they proposed a deep learning-based multi-model ensemble method with a 5-fold stacking and a five-layer neural network. They first applied k-nearest neighbor, support vector machines, decision trees, random forests and gradient boosting decision trees- five classification methods in the first stage individually. To derive the predictions from these methods, they used 5-fold cross-validation technique and averaged the predictions from the earlier stage. They applied a deep learning-based multi-model ensemble method for extracting insights from the datasets and by using the first-stage predictions as features, they had been able to reduce more generalization error than by training them in isolation. To overcome higher computational cost, they carried out a feature selection method in data preprocessing phase, and this resulted in improved prediction accuracy.[9]

Khamparia A. et al. in "A Novel Deep Learning-Based Multi-Model Ensemble Method for The Prediction of Neuromuscular Disorders" (2018) paper, adopted convolutional neural networks (CNN) as the ensemble method to assemble multiple classifiers. The fitness of the method was computed based on the ensemble outputs. They proved using the obtained results that the learning method utilized the clinical sample data more effectively.[10] Similar to other ensemble deep learning models, this increased computational cost. They performed feature selection techniques which reduced the running time and improved prediction accuracy the same as the previous paper mentioned. But because of the huge number of genes, the proposed algorithm could not find a way around the high-dimensional space issue.

In "Deep Unsupervised Clustering Using Mixture of Autoencoders" (2017) paper by Zhang, D. et al., their approach was to use a MIXture of AutoEncoders (MIXAE)- a separate autoencoder to model each data cluster, and thereby the entire dataset as a collection of autoencoders. To model the data cluster, they had to initialize the number of clusters. By training simultaneously with a mixture assignment network via a composite objective function, the autoencoders are proved to have low reconstruction error per manifold and cluster identification error. This kind of joint optimization has been shown to have good performance in other unsupervised architectures as well. Here, for each input data, the concatenated latent features are taken as input by the mixture assignment network. This outputs soft clustering assignments, and the mixture aggregation is done by combining the weighted reconstruction error together with proper regularizations.[11] Although they managed to improve performance on the unbalanced dataset over Deep Embedded Clustering (DEC), the accuracy for different handwritten digit datasets (i.e., MNIST) is lower than that of other methods such SC-EDAE.

Based on this literature research, it can be noted that some ensemble deep learning models are facing challenges with false negative predictions, overall accuracy, an optimal algorithm for global dataset and more. Whereas some models have been able to outperform and achieve better results and predictions in some artificial intelligence applications. Again, due to extra processing time and cost overheads compared with traditional ensemble learning, ensemble deep learning algorithms still needs more enhancement in some specific fields, where the time required to development and processing resources are usually restricted or the data to be processed is with large dimensionality.

3. Spectral Clustering with Ensemble of Autoencoders

3.1 Notation

In this paper, for notation, number of data points is n and the number of clusters is k. The objective of this project is to group the n number of data points into k clusters. For this the number of different autoencoders used is symbolized as m. X and $\hat{X}$ are the original input and reconstructed output.

3.2 Autoencoders

Autoencoders are the building block of this project. Moving forward with this section, I will introduce autoencoders and their contribution briefly.

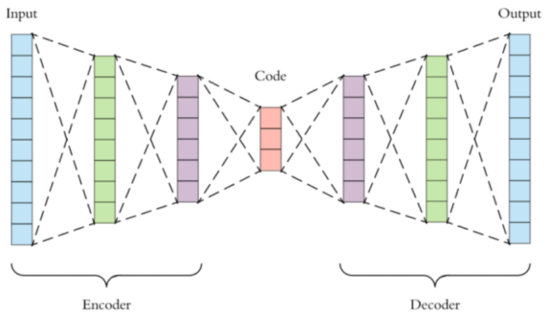

Figure 1: Basic concept of an Autoencoder [12]

An autoencoder is a special type of neural network. For example, if the data is an image, autoencoder tries to encode the image from higher dimension to lower dimension and then again to higher dimension same as the input image size. It tries to minimize the error by reconstructing the input image to a similar image with more information held in the intermediate layers.

The project autoencoder consists of three components:

- Encoder: In this part, the input data (here image) is encoded to lower dimension using multiple convolutional networks. By doing this, the input image is compressed into a latent space representation. The compressed image is the distorted version of the original image.
- Code: As shown in Figure 1, the middle layer between the encoder and decoder is the code part. In this layer, I used dense function to compress the previous layer input to multiple units. This layer is the bottleneck layer. It tries to hold as much information as possible without losing any important features of the original image.
- Decoder: This feedforward network is similar to the encoder, but the difference is that it tries to decompress the compressed data and tries to mimic the original input without losing any important features. For this process, I used Conv2DTranspose library function to have the same dimension of the original image.

Structure of the autoencoders used for this project are listed in table 1:

Table 1: Autoencoder structure

	Input Shape	Filters	Encoder output size
Autoencoder-1	28x28x1	32, 64, 128	Dense 10 units
Autoencoder-2		32, 64, 128, 64	
Autoencoder-3		16, 32, 64, 128	

3.3 Ensemble of Autoencoders

In this part, I designed multiple autoencoders with different values and different number of the filters. However, in all cases, the output dimension is the same as the input dimension.

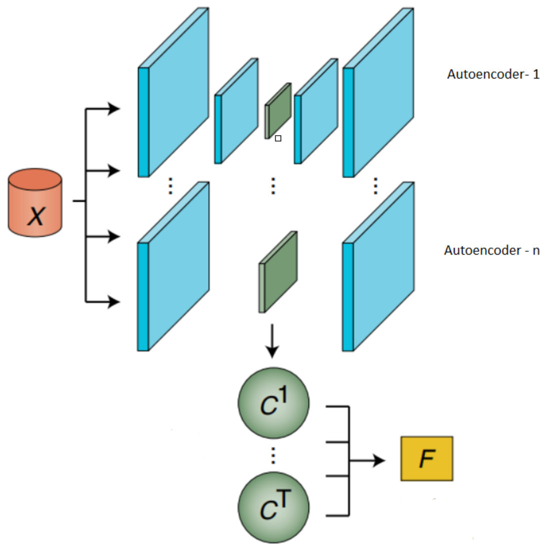

Figure 2: Ensemble of Autoencoders [7]

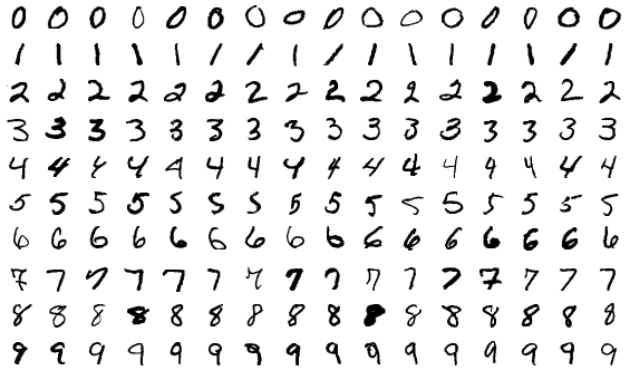

Figure 3: MNIST Dataset [13]

While training the autoencoders, the loss function used was mean squared error. The mean squared function measures the average of the squares of the errors. The equation of mean squared error is given by,

$$\text{Mean squared error} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \hat{X}_i\right)^2 \qquad (1)$$

In equation (1), X_i is the input image and $\widehat{X_\iota}$ is the reconstructed image found at the end of the decoder. Because of the square in equation (1), the result of the error is almost always positive. When the autoencoders are fully trained, the encoded parts are the input for the clustering. However, if the average or summation of encoded images are taken, the values would be somewhat scattered because in that case, the hyperdimensional values of different models are to be considered and each model embeds the input in their own ways.

To choose a final model for the encoded image, reconstructed part of every input is considered. Here, I do not take all images at once into consideration. Rather, every single image of training and test dataset are evaluated individually. The algorithm for this part which is later mentioned as outlier score is described below:

Algorithm

```
for image in range(all input images):
    for i in range(all autoencoders):
        reconstructed image[i] = autoencoder[i](image)
        mean squared error [i] =
            (reconstructed image – input image)$^2$
    autoencoder with min(mean squared error) += 1
```

Using the above-mentioned algorithm, the autoencoder with maximum outlier score is the final autoencoder model as shown in 2.

3.4 Spectral Clustering

Spectral clustering has become increasingly popular over the years due to its simple implementation and promising performance in many graph-based clustering. To perform spectral clustering, there are three main steps:

- Create a similarity graph between n data points
- Compute the first k eigenvectors of its Laplacian matrix to define a feature vector
- Run K-means on these features to separate objects into k clusters

So according to the steps, first the data is transformed from high dimensional feature space to similarity space. The data is projected into the new co-ordinate space which encodes information about how nearby data points are. The similarity transformation reduces the dimensionality of space and pre-clustering the data into orthogonal dimensions. The mapping to similarity space is facilitated by the creation and diagonalization of the similarity matrix. When k spatially separated and well-defined clusters are to be formed, the resulting similarity matrix is k-block diagonal and each block corresponds to a different cluster. Finally, the K-means part is run because the eigenvectors can be degenerative, and the clusters do not have to be so cleanly separated. The eigenvectors span the linear space defined by the clusters. However, the clusters could sit at any coordinates and are made sure that they are rotated 90 degrees from each other relative to the origin.

In the project, the encoded embeddings from the final chosen autoencoder are fed into the spectral clustering model. As the input of the spectral clustering, number of clusters is given. The output of the clustering method gives the result of the values in the different clusters.

In the project code, I used python as the programming language and tensorflow and keras as the library for training the autoencoder models, numpy library for different mathematical operations and sklearn library for spectral clustering.

4. Dataset

The dataset on which this project has been worked on is the MNIST dataset. MNIST (Modified National Institute of Standards and Technology database) is a large database of handwritten digits. The dataset contains 60,000 training images and 10,000 test images. The images are in black and white and the size of each image is set as 28 x 28 pixel as input of the autoencoders.[13] Few examples of the dataset are shown in figure 3.

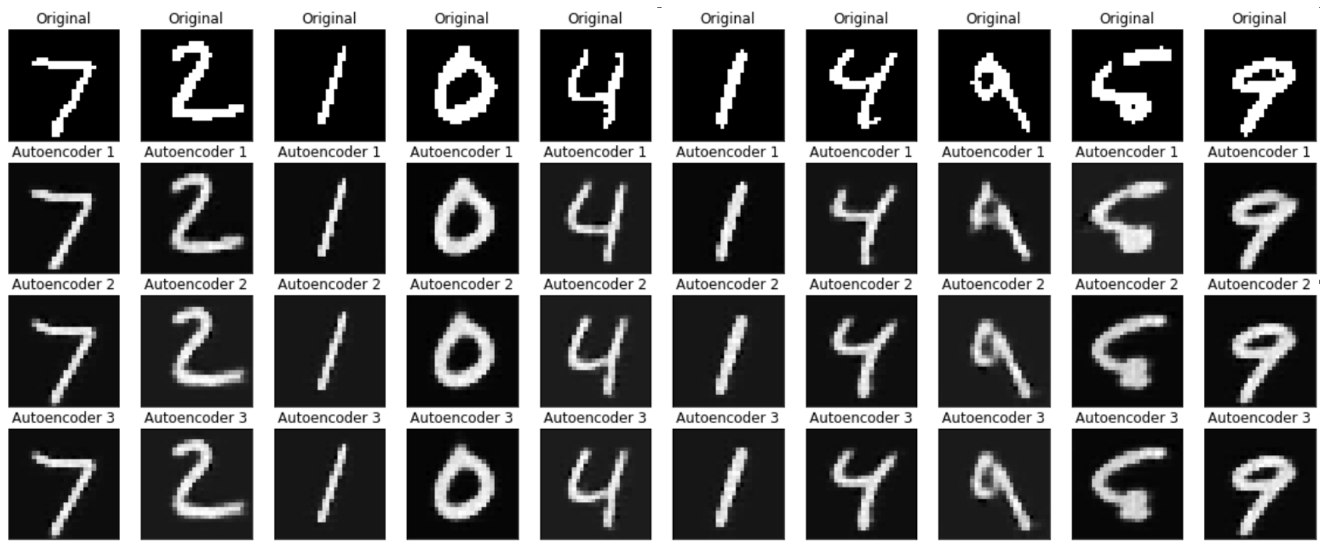

Figure 4: Reconstructed outputs of different autoencoders

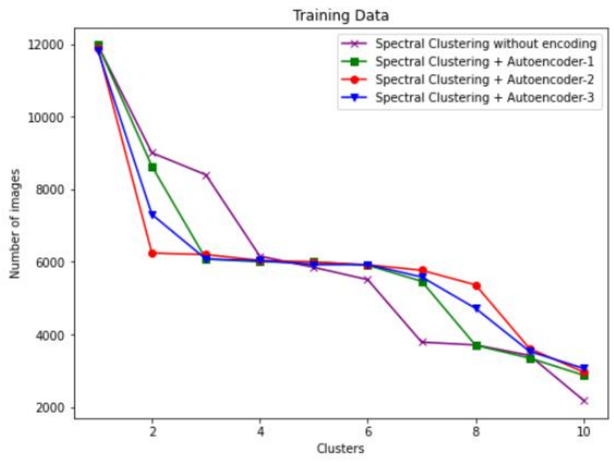

Figure 5: Comparison of results with and without autoencoders on training data

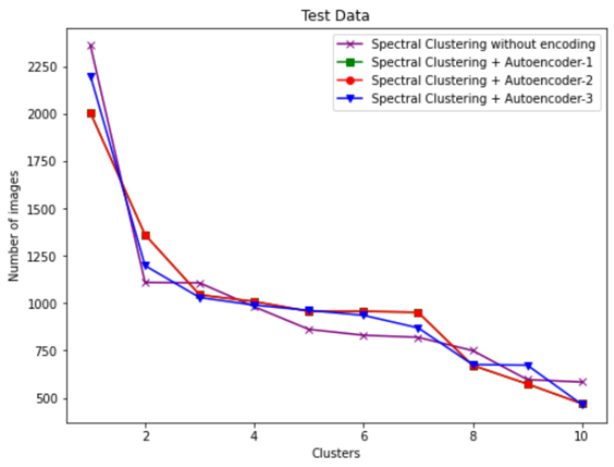

Figure 6: Comparison of results with and without autoencoders on test data

5. Evaluation & Discussion

The scores of different autoencoders are shown in table 2 based on mean squared error metric.

Table 2: Outlier Score

	Autoencoder- 1	Autoencoder- 2	Autoencoder- 3
Training Data	13034	33603	13363
Test Data	3970	3533	2497

As we can see from the Table 2, the best score on the training data is found by autoencoder-2. Although autoencoder-1 performed better result on test data, autoencoder-2 is the final model from the ensemble of models found while training.

Now when spectral clustering is performed on training and test data, autoencoder-2 indeed performed better than other autoencoders. To compare the results with spectral clustering algorithm on original data, the result is shown in the figure 5 and 6 for training and test data, respectively. Also, final clustering is shown in Figure 7. Images from similar cluster is shown in the same row.

From figure 5, it is visible that autoencoder-2 performed better results while clustering training data. Seven clusters have around 6000 images for the training data and one of the clusters has about 12000 images while other two have less than 4000 images. From figure 7, we can note that the clustering algorithm has difficulty in recognizing images for handwritten digits of 3, 5, 8 which are the three clusters with inaccurate values. Similar kind of result can be noticed for test data in figure 6. Five clusters have around 1000 images per cluster. From figure 5 and 6, the results of other

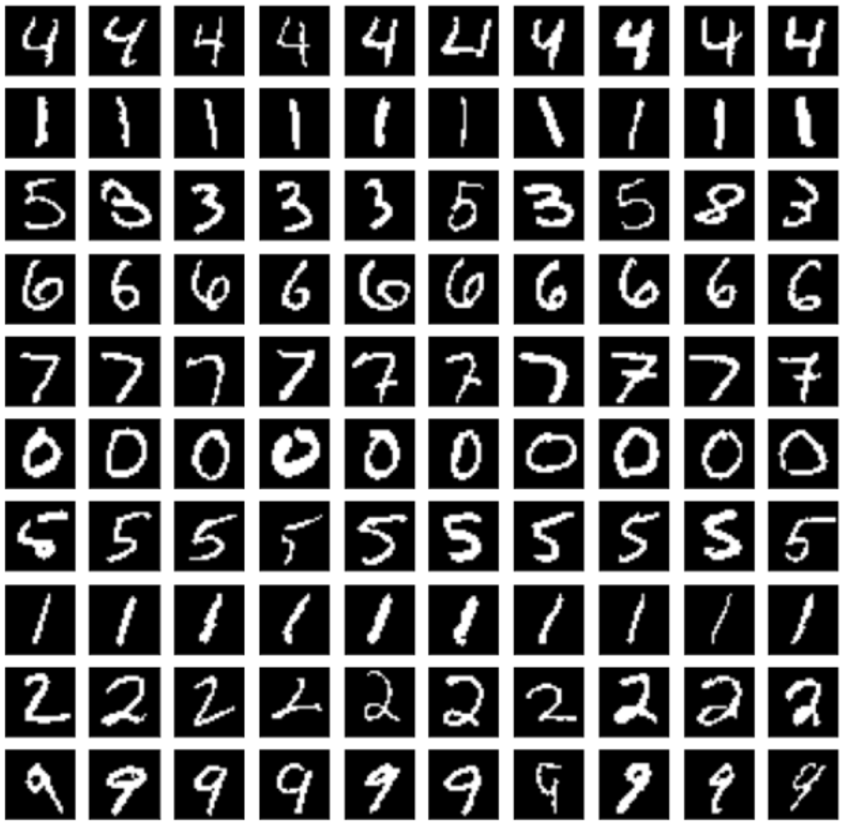
Figure 7: Final clusters on test data

autoencoders are quite close if not same. For test data, the result of autoencoder-1 and autoencoder-2 is same, and they perform same curve, even though on test data, the outlier score was better for autoencoder-1. However, autoencoder-2 is the one which performed stability over both training data and test data.

I compared my experimental results with spectral clustering algorithm on the unaltered original data. As visible from figure 5 and 6, spectral clustering on original data performs worse and the number of images per clusters are scattered. This method also took over 150 minutes to complete on a GPU, while my experiment on ensemble of clusters and spectral clustering on the best model took less than 90 minutes (for 50 epochs of training for each autoencoder). The overall accuracy found for my experiment in an unsupervised manner is close to 80%.

6. Conclusion & Future Work

From the experiment, it is evident that spectral clustering on encoded data brings out better performance and better results rather than using spectral clustering on original data. When there are multiple autoencoders to try on, it is advisable to perform ensemble of them as evident from the experiment results.

There are some aspects of this experiment where there is scope for improvement. The autoencoders used here are simple convolutional autoencoders. There is maximum 4 convolutions done for simplicity at the encoder. More accuracy and better result could be found if complex autoencoder is used. Also, variational autoencoder could be used to compare results. Again, the metric for outlier score here was mean squared error which does not necessarily give any information how accurate the encoded value is. Rather it just gives information how much different the input image and the reconstructed image are. So, it would be better if any other metric for outlier score could be used. Finally, from the experiment, it is safe to say that spectral clustering with ensemble of autoencoders is preferable.

References

[1] https://www.xplenty.com/blog/structured-vs-unstructured-data-key-differences/
[2] https://www.investopedia.com/terms/d/deep-learning.asp
[3] https://www.analyticsvidhya.com/blog/2018/05/essentials-of-deep-learning-trudging-into-unsupervised-deep-learning/
[4] https://www.mygreatlearning.com/blog/introduction-to-spectral-clustering/
[5] https://www.edureka.co/blog/autoencoders-tutorial
[6] Dietterich, TG. “Ensemble Methods in Machine Learning” (2000)

[7] Cao, Y. et al. "Ensemble deep learning in bioinformatics" Review Paper (2020)
[8] Geddes et al. “Autoencoder-based cluster ensembles for single-cell RNA-seq data analysis” (2019)
[9] Xiao, Y. et al. “A deep learning-based multi-model ensemble method for cancer prediction” (2018)
[10] Khamparia A. et al. “A Novel Deep Learning-Based Multi-Model Ensemble Method for The Prediction of Neuromuscular Disorders” (2018)
[11] Zhang, D. et al. “Deep Unsupervised Clustering Using Mixture of Autoencoders” (2017)
[12] https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798
[13] https://en.wikipedia.org/wiki/MNIST_database