Visualizing Data using t-SNE

Team Number: 55 | Team Bash Party

- Keshav Bajaj (2019115010)
- Tejasvi Chebrolu (2019114005)
- Naman Ahuja (2019101042)
- B Vaibhaw Kumar (2019112021)

Problem Statement

Visualization of high-dimensional data is an important problem in many different domains and deals with data of widely varying dimensionality. Cell nuclei that are relevant to breast cancer, for example, are described by approximately 30 variables (Street et al., 1993), whereas the pixel intensity vectors used to represent images or the word-count vectors used to represent documents typically have thousands of dimensions.

Dimensionality Reduction methods convert the high-dimensional data set $X = \{x_1, x_2, ..., x_n\}$ into two or three-dimensional data $Y = \{y_1, y_2, ..., y_n\}$ that can be displayed in a scatterplot. The aim of dimensionality reduction is to preserve as much of the significant structure of the high-dimensional data as possible in the low-dimensional map. Various techniques for this problem have been proposed that differ in the type of structure they preserve.

We plan on implementing **t-SNE**, a way of converting a high-dimensional data set into a matrix of pairwise similarities, for visualising the resulting similarity data. **t-SNE** is capable of capturing much of the local structure of the high-dimensional data very well, while also revealing global structure such as the presence of clusters at several scales. It is also **non-linear** which means that it can separate data that cannot be separated by any straight line.

- **Principal Component Analysis**: is used to explain the variance-covariance structure of a set of variables through linear combinations. It is often used as a dimensionality reduction

technique. It is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.

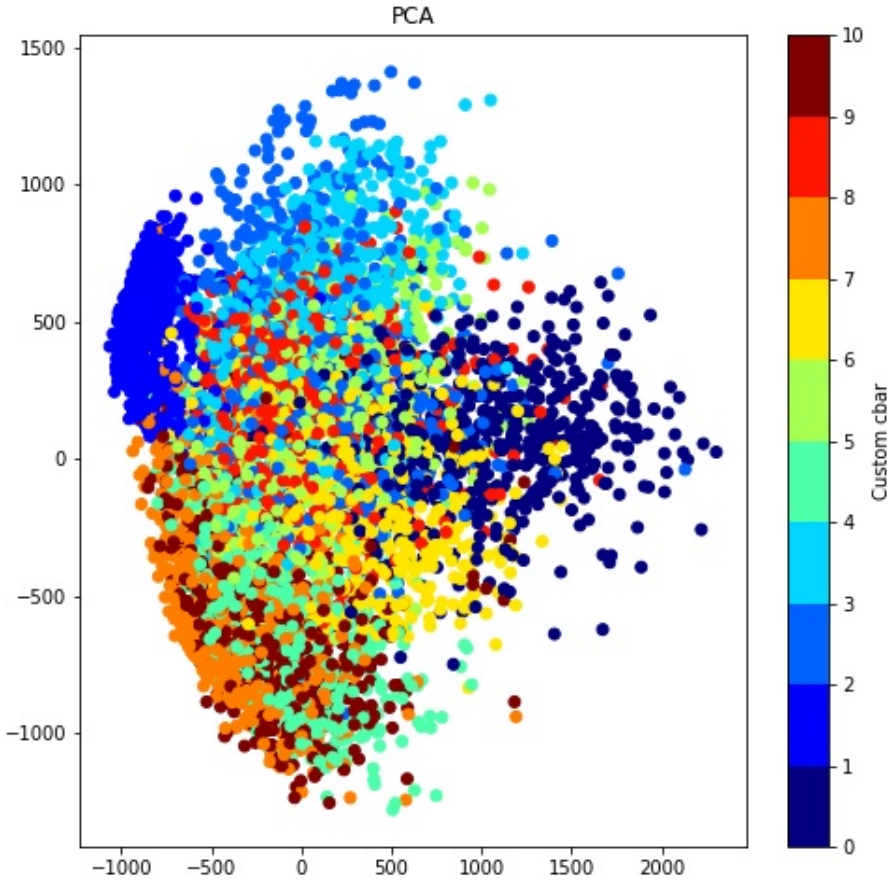
- **Sammon Mapping**: **Sammon mapping** or **Sammon projection** is an algorithm that maps a high-dimensional space to a space of lower dimensionality by trying to preserve the structure of inter-point distances in high-dimensional space in the lower-dimension projection.
- I**sometric Mapping**: **Isomap** is a non-linear dimensionality reduction method. It is one of several widely used low-dimensional embedding methods. Isomap is used for computing a quasi-isometric, low-dimensional embedding of a set of high-dimensional data points. The algorithm provides a simple method for estimating the intrinsic geometry of data manifold based on a rough estimate of each data point's neighbours on the manifold. Isomap is highly efficient and generally applicable to a broad range of data sources and dimensionalities.

- **Understanding MNIST and Olivetti Face Datasets**: We used two datasets to try out the above-mentioned DR techniques to understand their working.
    - MNIST Dataset: The MNIST(Modified National Institute of Standards and Technology) database consists of handwritten digits. It has a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centred in a fixed-size image.
    - Olivetti Faces Dataset: This dataset contains a set of images of faces taken between April 1992 and April 1994 at AT&T Laboratories Cambridge. There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling / not smiling), and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

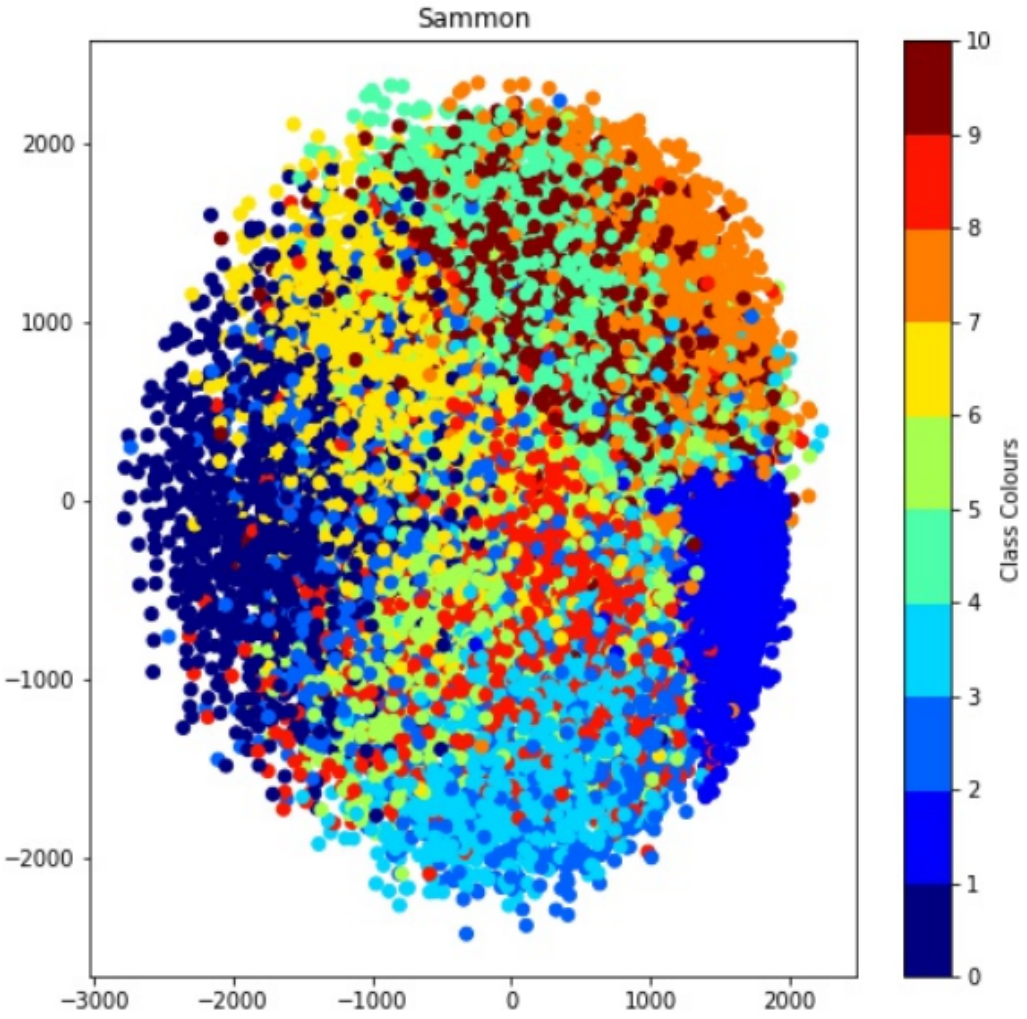
Work Done

- Ran PCA, Sammon, Isometric Mapping on MNIST and Olivetti Faces datasets for dimensionality reduction.
    - MNIST and Olivetti Faces

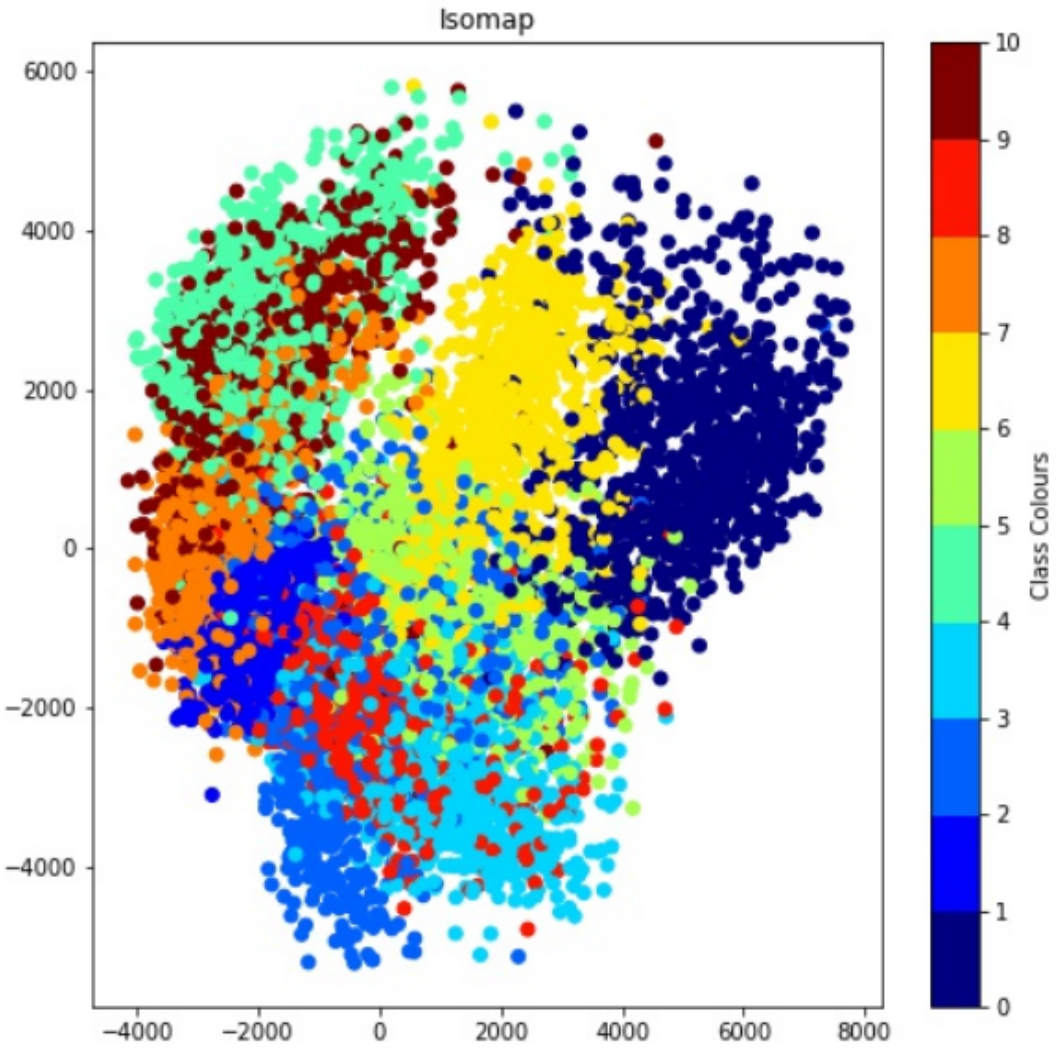
- Reduced the initial 784 dimensions to 2 using Principal Component Analysis (Visualization shown below)
- MNIST Dataset (subsample of 5000 points reduced to 2 dimensions using PCA)
- Reduced the initial 784 dimensions to 30 using PCA
    - PCA after reducing dimensions from 784 to 2



- Sammon Mapping to reduce 30 dimensions to 2 for visualisation

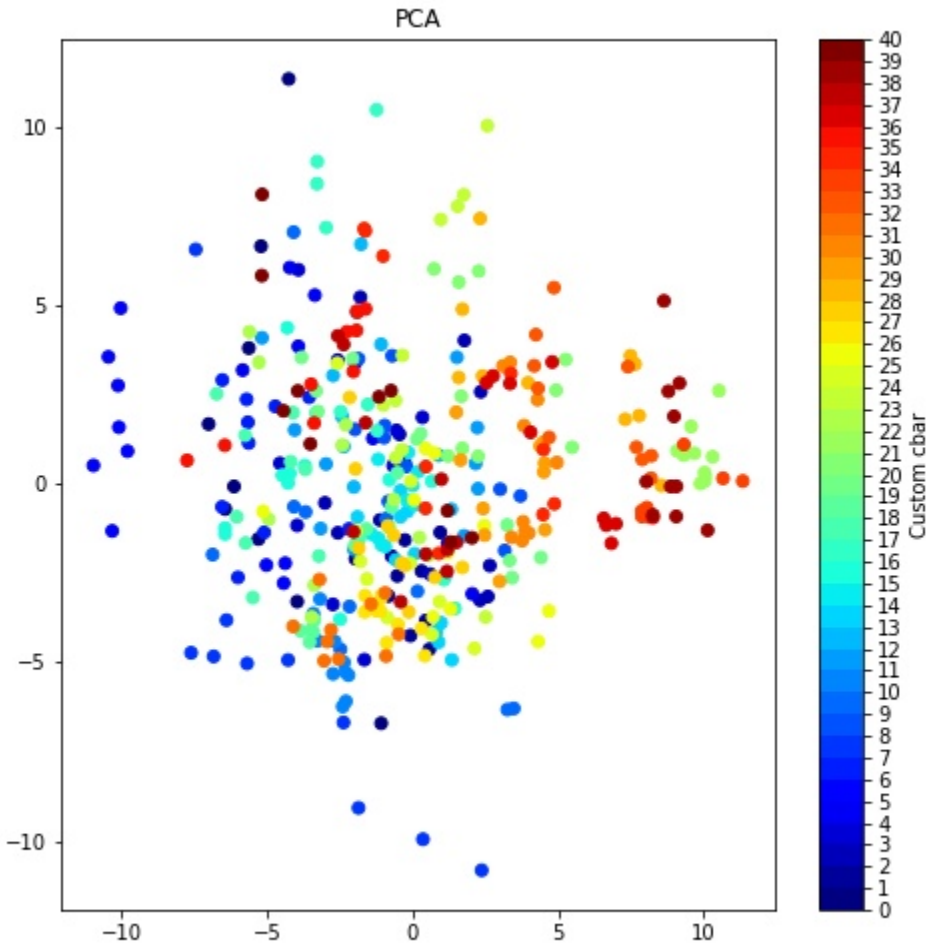


- Isometric Mapping to reduce 30 dimensions to 2 for visualization

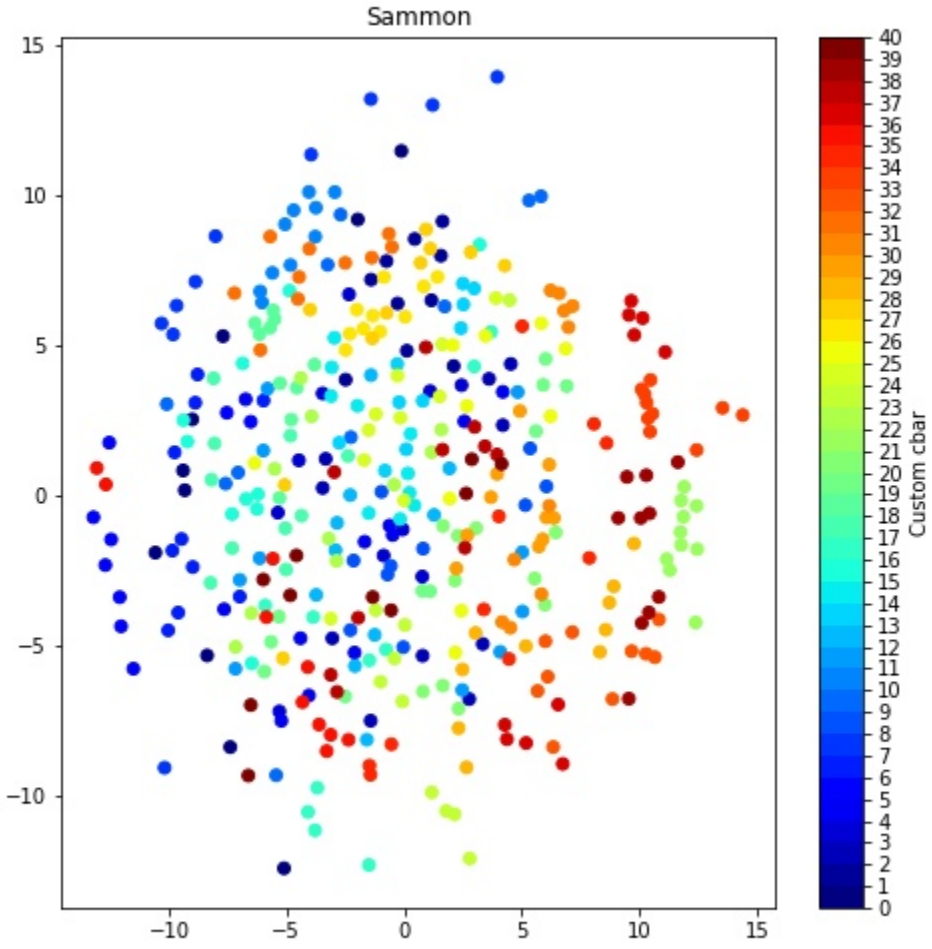


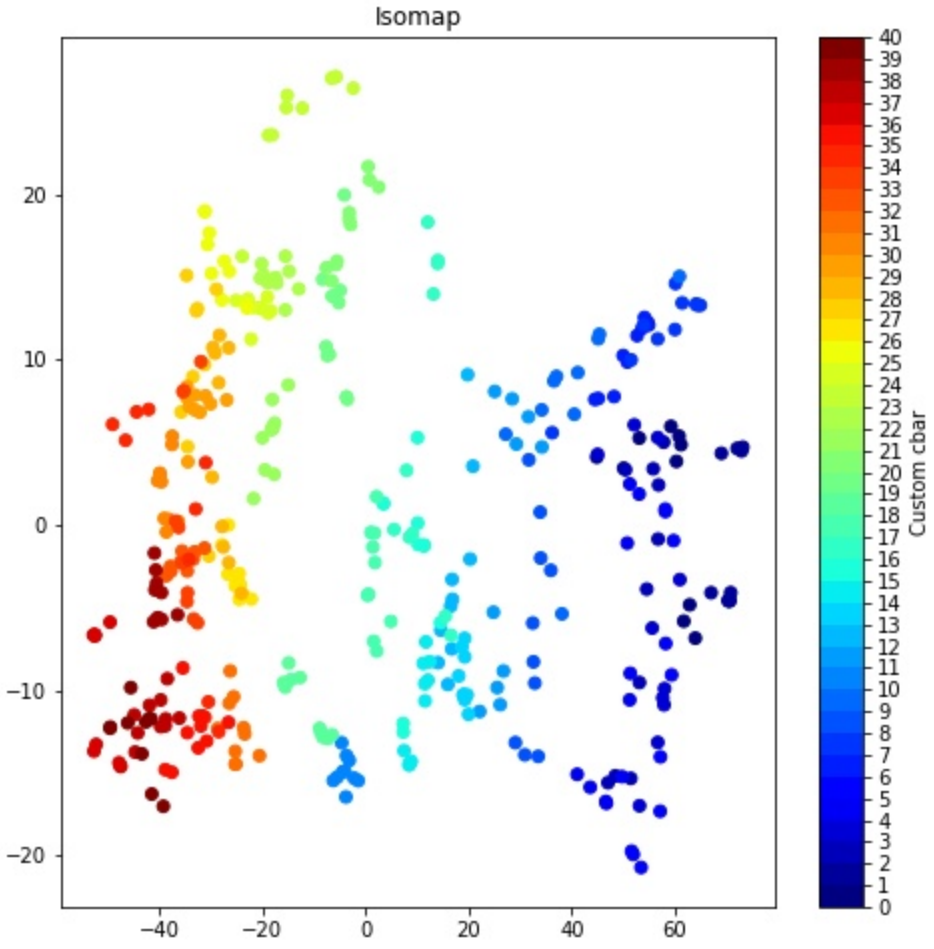
Olivetti Dataset

- Ran PCA on Olivetti Dataset to reduce 4096 dimensions to 2



- Ran PCA to reduce 4096 dimensions to 30. Then ran Sammon and isometric mapping for visualization.


Sammon
15
10
5
0
−5
−10
−10
−5
0
5
10
15
Custom cbar
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0



SNE- Stochastic Neighbor Embedding Algorithm:

When given a dataset X, with N data points, where each datapoint x i has D dimensions, SNE (or t-SNE as well) will reduce each data point to d(for simplicity, d=2) dimensions.

In SNE, the Euclidean distance between data points is converted into conditional probabilities that represent similarities.

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-||x_i - x_k||^2 / 2\sigma_i^2\right)}$$

(the will be discussed later)

So basically, The probability of point **xj** being a neighbour of point xi is proportional to the distance between them. Note- pi|i=0 for all i as we are not concerned about whether or not any point is a neighbour of itself or not.

Now, let Y be an Nx2 matrix which is a 2D representation of X, and distribution q can be constructed as per the construction of p.

$$q_{j|i} = \frac{\exp\left(-||y_i - y_j||^2\right)}{\sum_{k \neq i} \exp\left(-||y_i - y_k||^2\right)}$$

(Note that here we are not using)

The ultimate purpose is to select points in Y that are comparable to p in the generated conditional probability distribution q. This is accomplished by minimizing a cost: the KL-divergence between the two distributions and be defined as given below:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

This cost needs to be minimised, and here, we will be using gradient descent and therefore only its gradient with respect to the 2D representation of Y is needed.

Perplexity

Perplexity is defined as follows for any row of the conditional probabilities matrix P:

$$Perp(P_i) = 2^{H(P_i)}$$

where H(Pi)is the Shannon Entropy of Pi in bits:

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$$

Perplexity is a parameter that we set in SNE (and t-SNE) (usually between 5 and 50). The i's are then set so that the perplexity of each row of P is equal to our desired perplexity – the parameter

we selected. With entropy, perplexity increases, thus, for higher perplexity, pj|i(for any given i) should be as similar to each other as possible, that is, the probability distribution Pishould be as flat as we can make it(as higher entropy means flatter probability distribution- the probability of the majority of the elements in the distribution are similar). This can be achieved by making increasing the i, as when divided by larger iprobability distribution gets closer to having all probabilities equal to 1/N.

So, if we desire more perplexity, we will increase our i's, which will lead the conditional probability distributions to flatten. If we consider xi and xj as neighbours if pj|i is less than a given probability threshold, this effectively increases the number of neighbours each point possesses. This is why the perplexity parameter is sometimes equated to the number of neighbours each point is thought to have.

Symmetric SNE

We minimise a KL divergence over joint probability distributions with entries pij and qij, rather than conditional probabilities pi|j and qi|j, in Symmetric SNE. Each qij in a joint distribution is defined by:

$$q_{ij} = \frac{\exp\left(-||y_i - y_j||^2\right)}{\sum_{k \neq l} \exp\left(-||y_k - y_l||^2\right)}$$

We can simply set

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}$$

instead of analogous distribution, to avoid problems related to outlier x points.

These newly defined points can be easily obtained-- the joint p is just (P+PT)/2N, where P is the conditional probabilities matrix with (i,j)'th entry pj|i.- We can generate the negative squared euclidean distances matrix from Y, exponentiate it, then divide all entries by the entire sum to estimate the joint q. After we get our joint distributions p and q, we can use-

to update the i'th row of the low dimensional Y.

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

We can update yi as we perform the gradient descent after finding the gradients, through the equation given below-

$$y_i^t = y_i^{t-1} - \eta \frac{\partial C}{\partial y_i}$$

t in t-SNE

Only the way we define the joint probability distribution matrix Q, which includes entries qij, differs in Symmetric SNE to t-SNE, and this can be seen below-

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}}$$

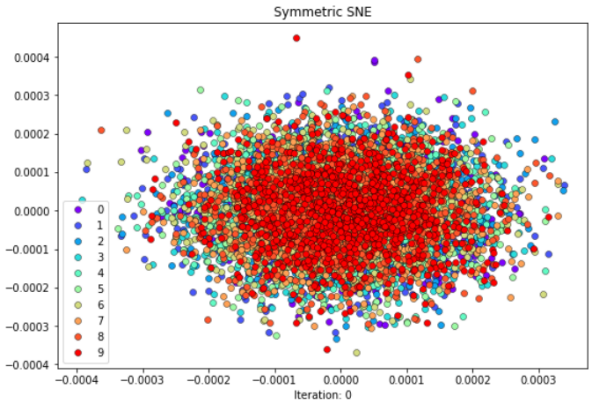
The algorithm is nearly invariant to the low-dimensional mapping's general scale. As a result, the optimisation works in the same way for very far apart points as it does for points that are closer together, because the numerator approaches an inverse square law over large distances in low-dimensional space.

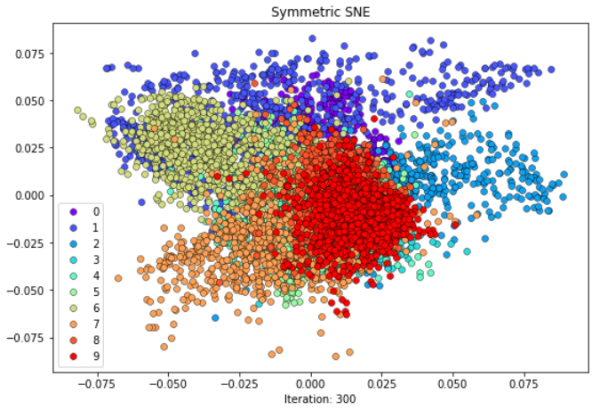
This solves the 'crowding problem': When we try to describe a high-dimensional dataset in two or three dimensions, it becomes impossible to distinguish between local data points and moderately-distant data points — everything becomes packed together, and the natural clusters in the dataset are prevented from being separated. The gradient derived in the t-SNE paper is as follows-

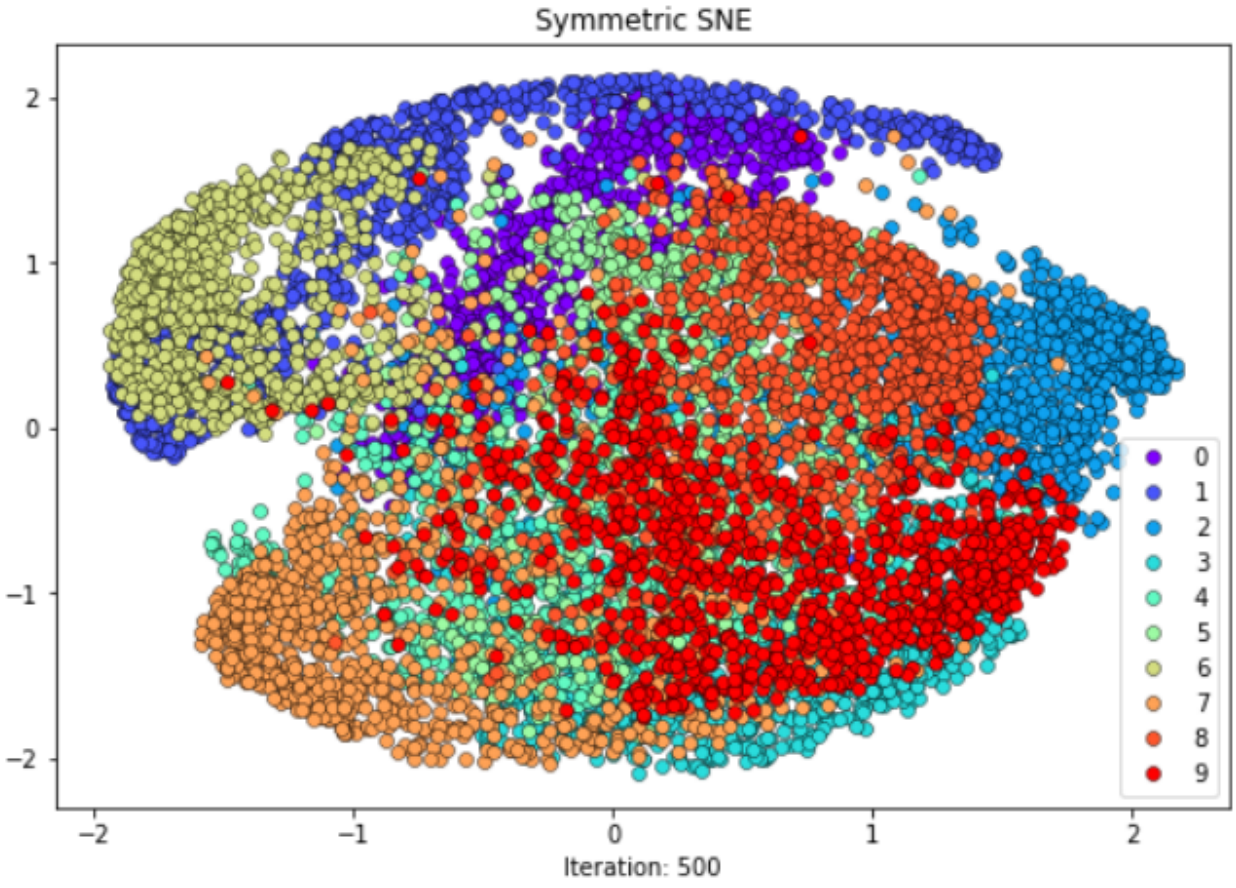
$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + ||y_i - y_j||^2\right)^{-1}$$

MNIST Dataset

Iterations

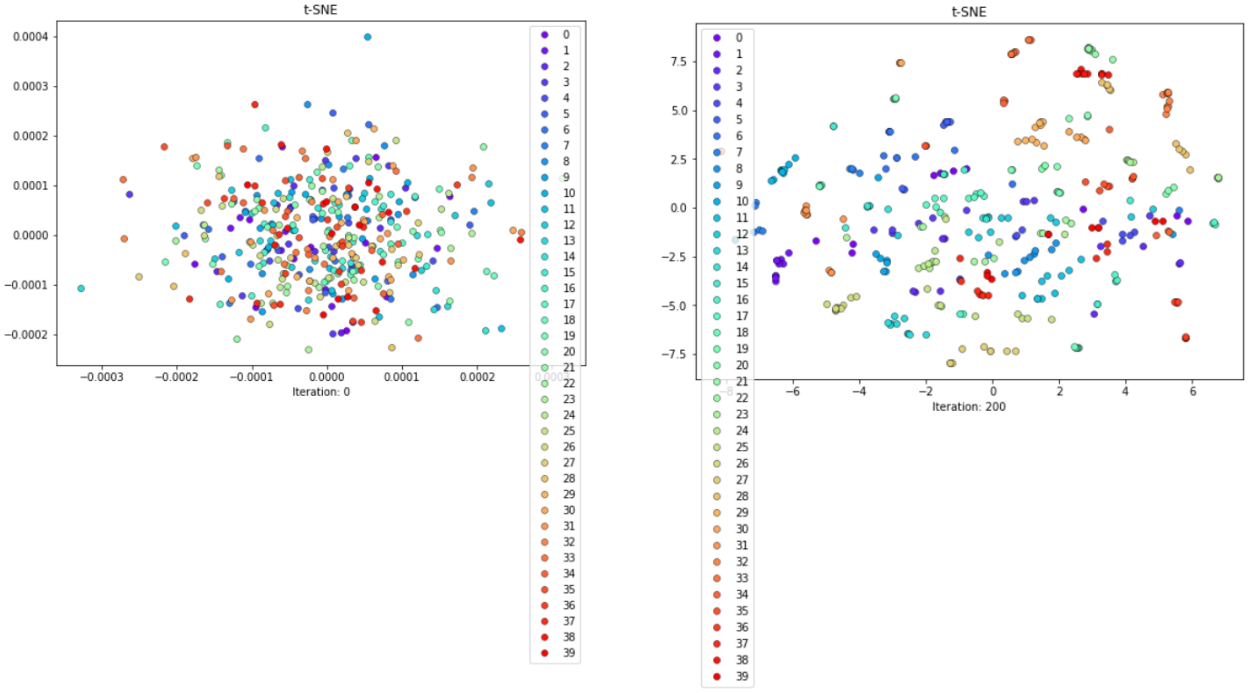


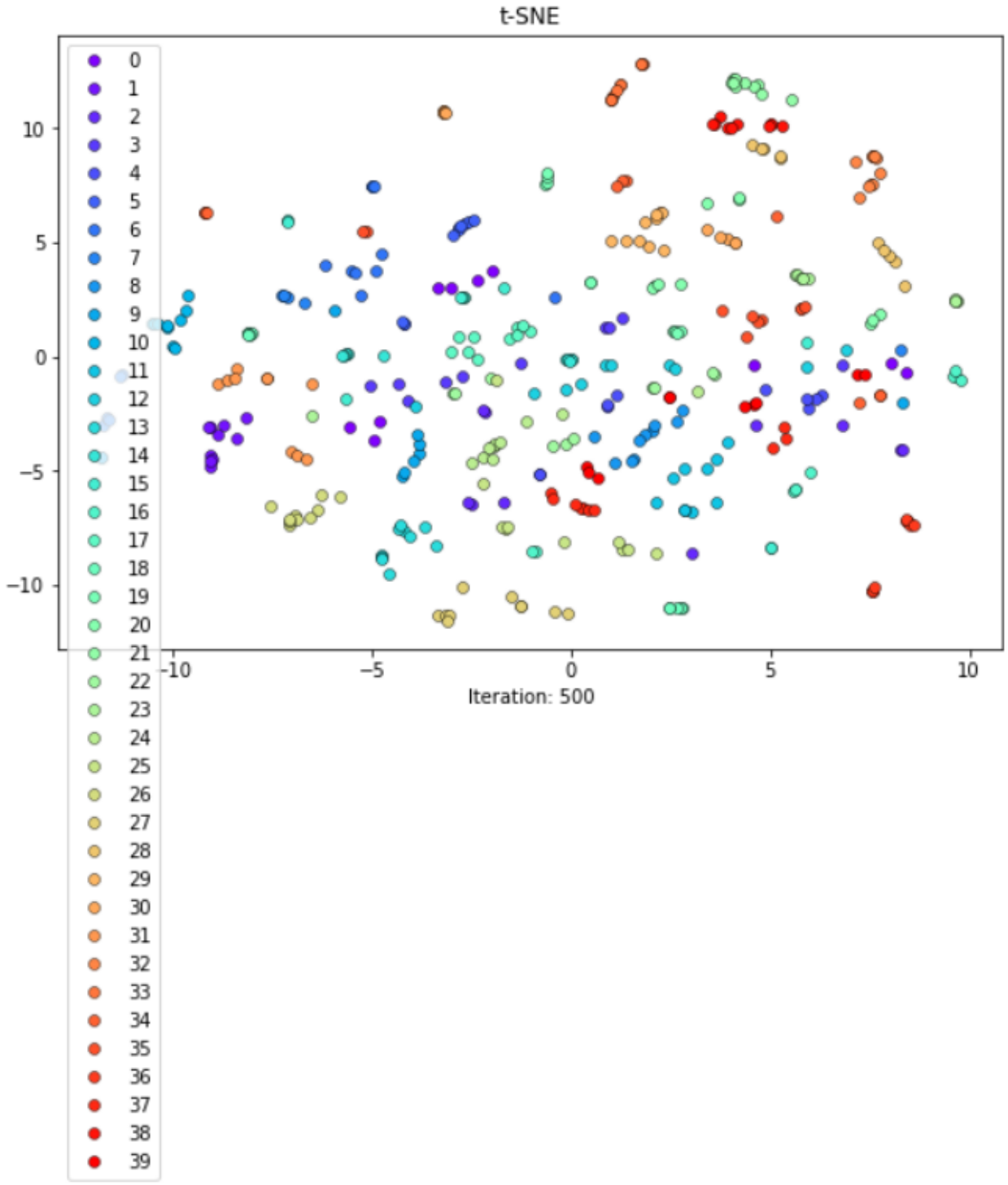




Olivetti Dataset

Iterations





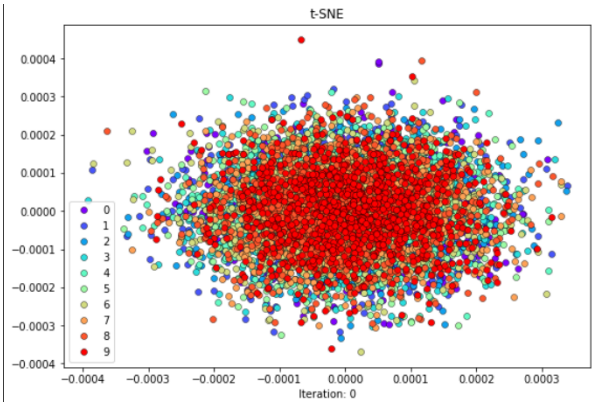
T-SNE:

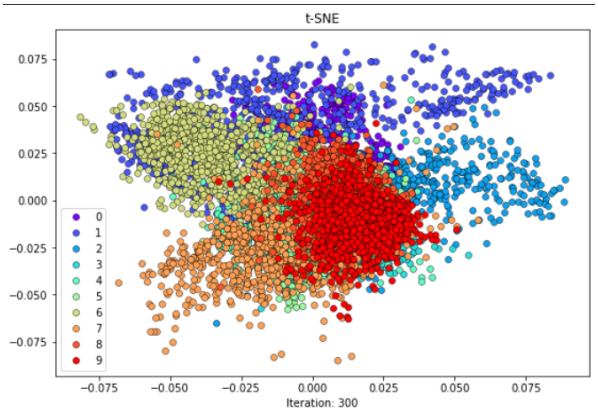
t-Distributed Stochastic Neighbor Embedding (t-SNE) is an algorithm in Machine Learning to perform dimensionality reduction on high dimensional data and visualize it in 2D or 3D, which is much easier for humans to understand.

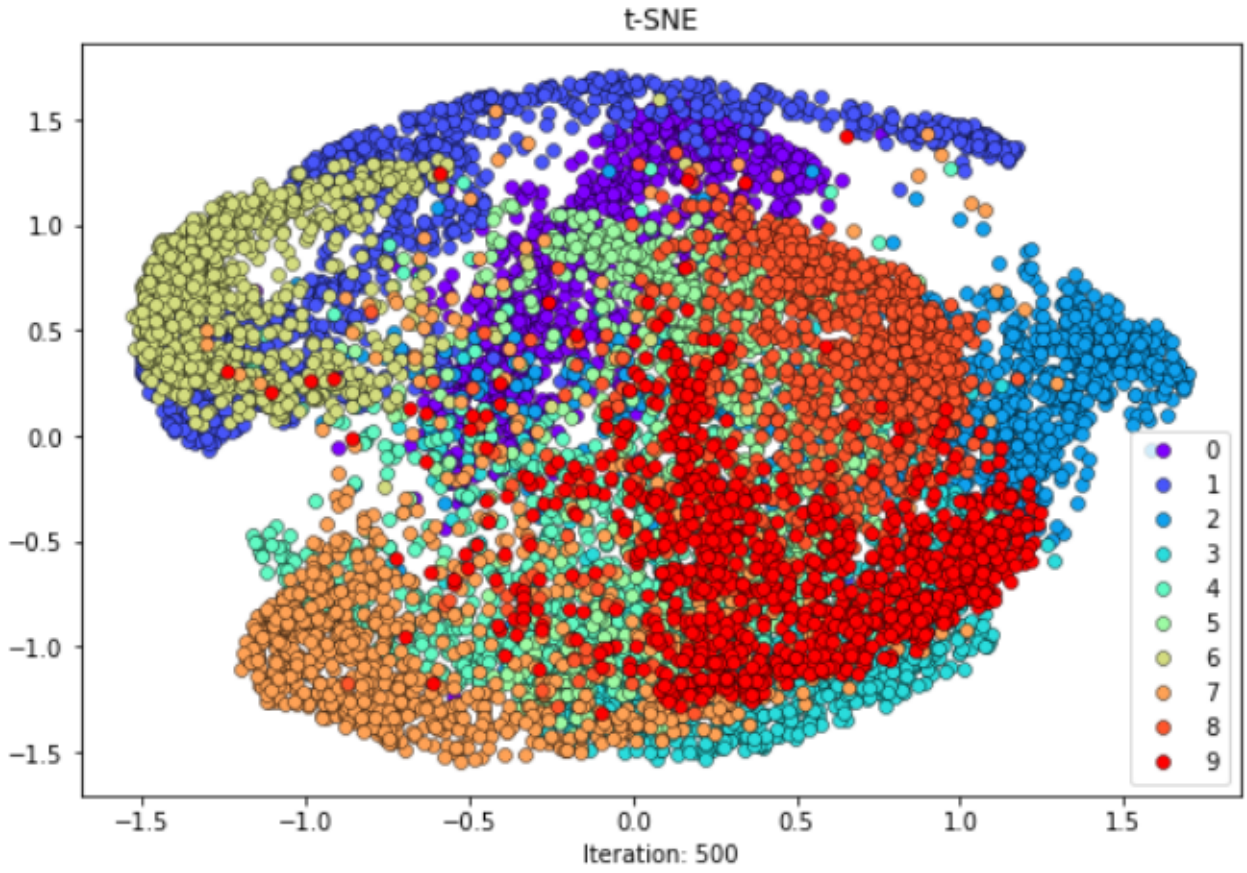
Taking MNIST for example, which is a dataset that contains 28x28 pixel handwritten images of digits from 0 to 9, that means that the data is in 784-dimensional space, but with t-SNE, it can be reduced to 2D space.

MNIST Dataset

Iterations

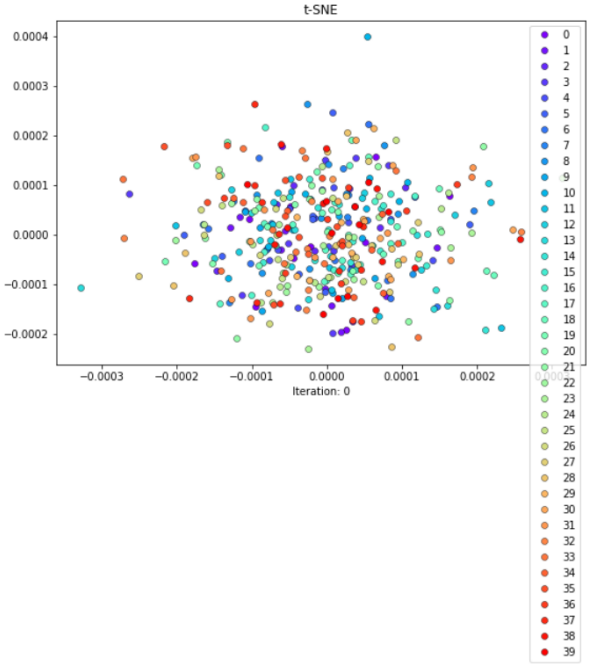


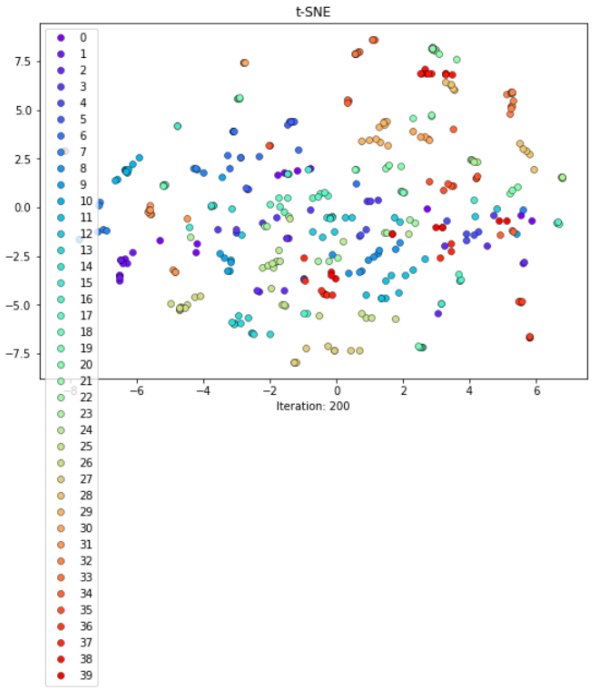


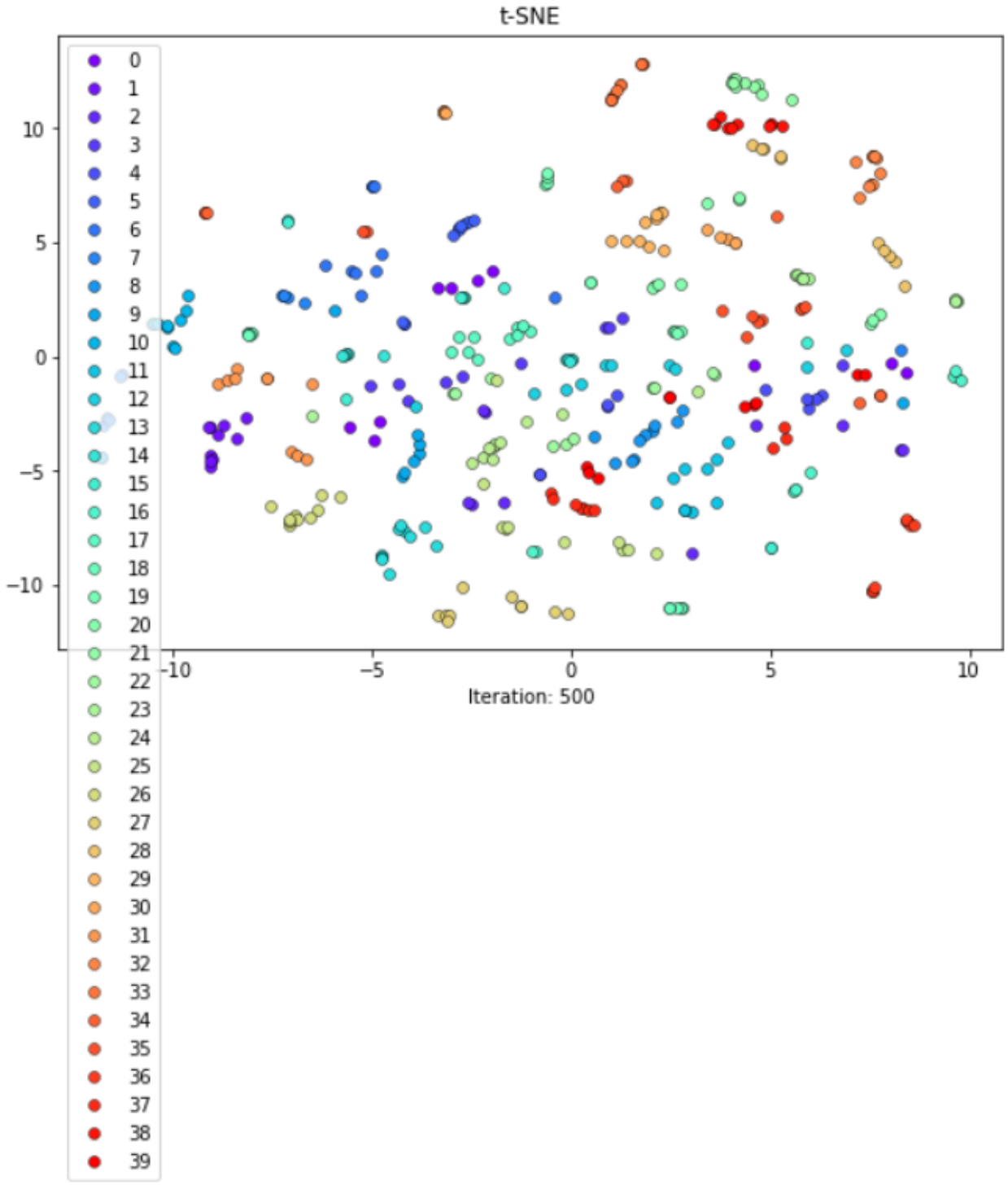


Olivetti Dataset

Iterations


t-SNE
Iteration: 0


t-SNE
Iteration: 200



References:

1. van der Maaten, Laurens & Hinton, Geoffrey. (2008). Visualizing data using t-SNE. Journal of Machine Learning Research. 9. 2579-2605.

2. *Statquest: T-sne, clearly explained*. (n.d.). Retrieved 7 November 2021, from https://www.youtube.com/watch?v=NEaUSP4YerM
3. Erdem (burnpiro), K. (2020, April 22). *T-sne clearly explained*. Medium. https://towardsdatascience.com/t-sne-clearly-explained-d84c537f53a