t-Distributed Stochastic Neighbor Embedding

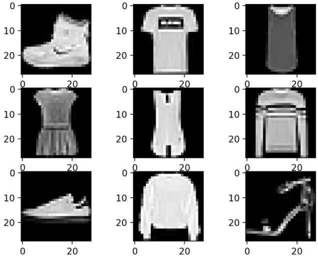
Kamila Bobkowska Patryk Krawiec

Adam Mickiewicz University

June 2020

Introduction

Our task was to implement **T-distributed stochastic neighbor embedding - t-SNE** and test it on a dataset of our choice. We chose a **MNIST_784** dataset with pictures of numbers and **Fashion-MNIST** a dataset with pictures of different types of clothes.



Recall - PCA

In order to use t-SNE one needs to understand **PCA**. PCA is a method to reduce dimensionality but preserve the generality(the most important features). It uses mathematics and linear algebra to do so as opposed to t-SNE which uses probability calculations.
In general, PCA checks which parameters are the most important for a given dataset.

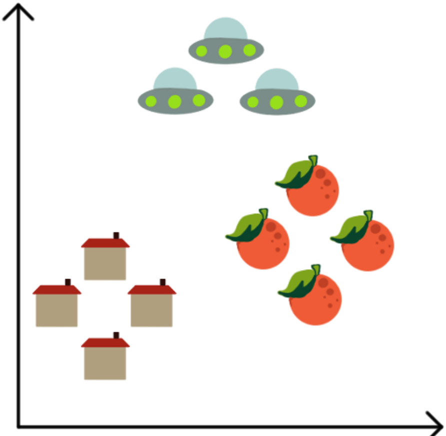
Recall - PCA

In more detail:

1. First we need to center our data around the origin. We can do so by subtracting the mean of each variable.
2. Next, we calculate the covariance matrix. Once we have that we calculate the eigenvectors and eigenvalues of this matrix.
3. Afterwards we normalize each of the orthogonal eigenvectors.
4. In the end, the obtained orthogonal eigenvectors, which represent how much each feature matters so we can easily see how much information we will lose by omitting it.

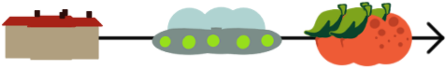
How does t-SNE work?

Let us create a simple 2D example:



How does t-SNE work?

How does t-SNE work?



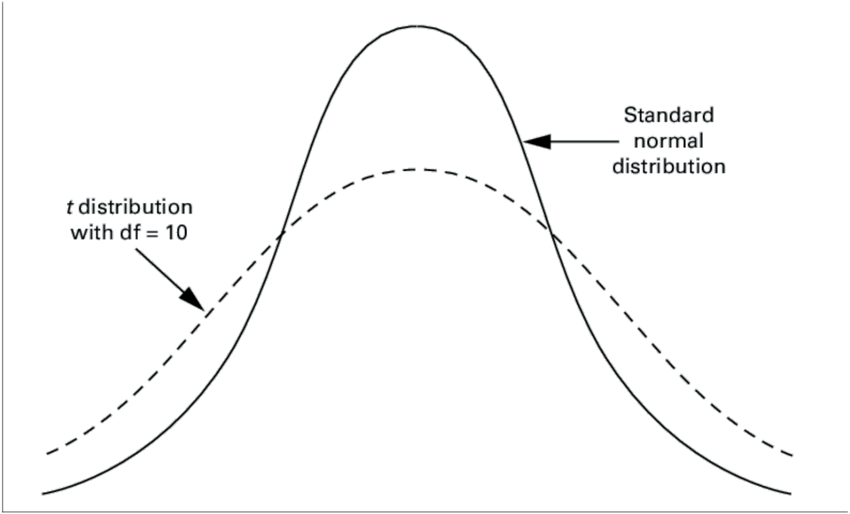
t-SNE in detail...

t-SNE is an unsupervised non-linear technique for representing highly dimensional data in a lower dimension while preserving original data similarity. It is mostly used for visualization of data and is especially useful with artificial intelligence and data exploration.

t-SNE an important note

t-SNE uses a t-distribution (hence the t) instead of normal distribution because we need to take into account points that are further away from the selected point. Due to the shape of the t-distribution graph change of importance diminishes less rapidly than in a normal distribution.

t-SNE an important note



t-SNE in detail...

In more detail:

1. In order to cluster the data in a lower dimension we take a single point and calculate pairwise the probability that this point and all the other points are in the same cluster proportional to the distance. We do it for all the points and end up with a matrix of probabilities.

t-SNE in detail...

2. Next, we calculate the probabilities in a lower dimension similarly as we did before.

t-SNE in detail...

3. After that we proceed to minimize the differences between the results we got in point 1 and 2 which in fact, minimizes the sum of **Kullback-Leibler divergence** using gradient descent.

Kullback-Leibler divergence

It is worth mentioning what is the **Kullback-Leibler divergence** or **relative entropy**. It helps us measure how much information we will lose once we approximate one probability distribution with another one.

$$D_{KL}(p||q) = \sum_{i=1}^{N} p(x_i) \cdot (\log p(x_i) - \log q(x_i)) \tag{1}$$

Our program

The code for fetching datasets:

```
mnist = fetch_openml('Fashion-MNIST') #or MNIST_784
X = mnist.data / 255.0
Y = mnist.target
```

The code responsible for PCA:

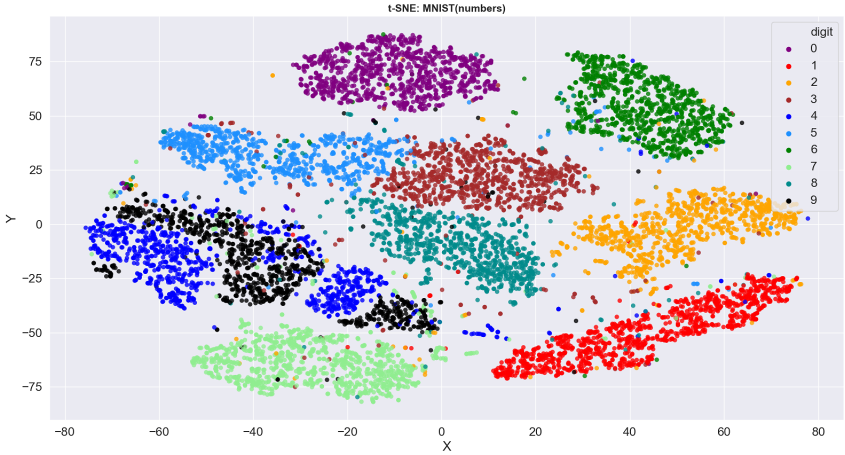
```
pca_n = PCA(n_components=50)
pca_result_n = pca_n.fit_transform(x_subset)
```

The code responsible for t-SNE:

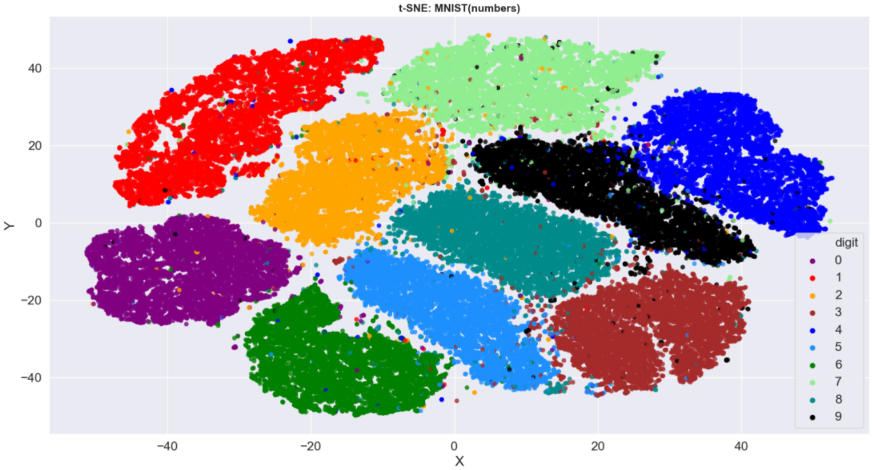
```
pca_tsne = TSNE(random_state=RS).fit_transform(pca_result_n)
```

Our program - MNIST_784

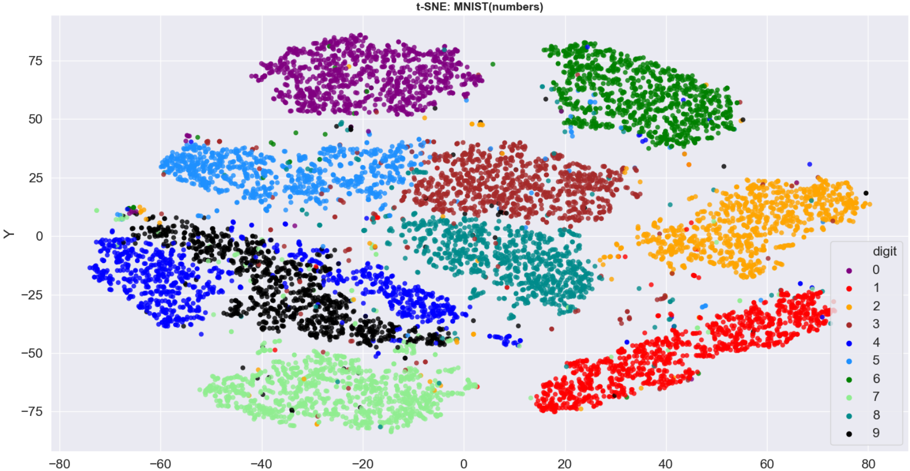
Our program - MNIST_784



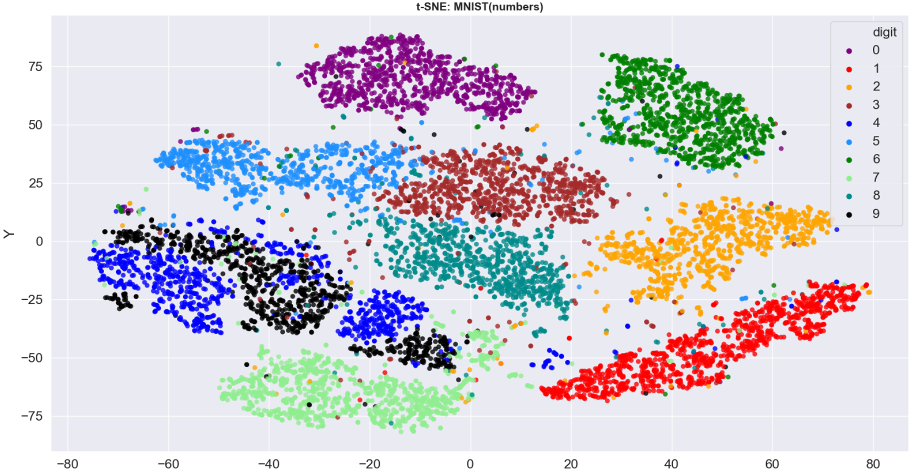
Our program - MNIST_784



Our program - MNIST_784 changing the number of PCAs - decreased

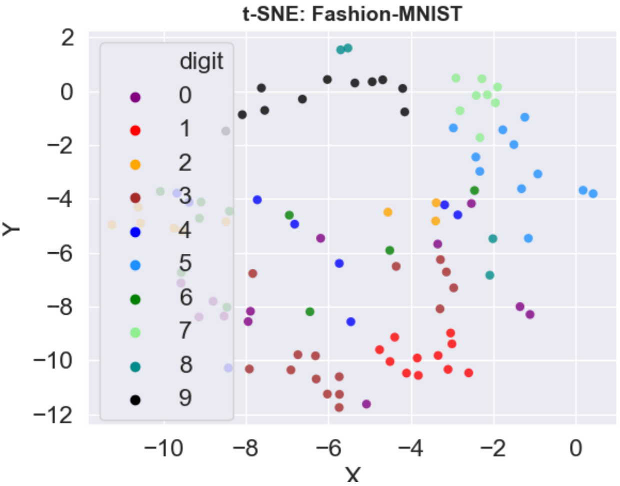


Our program - MNIST_784 changing the number of PCAs - increased

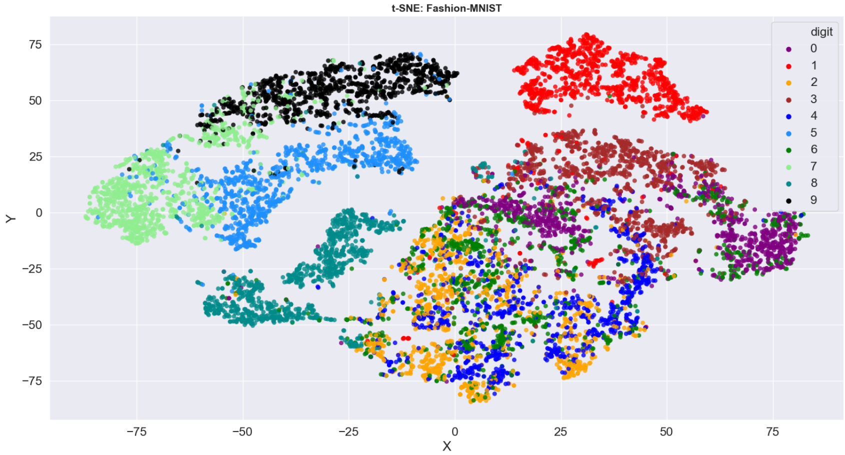


Our program - MNIST_784 changing the number of PCAs - increased

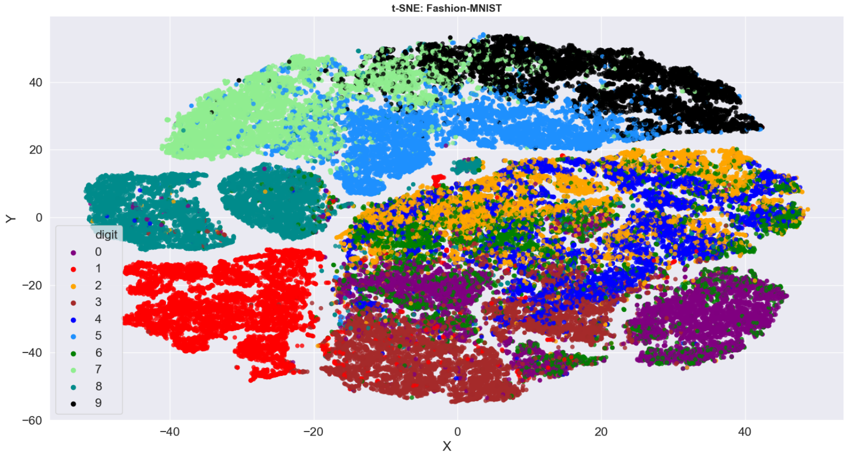
Our program - Fashion-MNIST



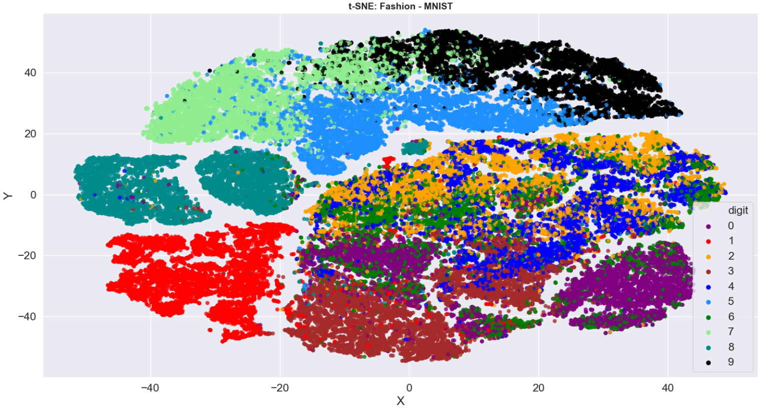
Our program - Fashion-MNIST



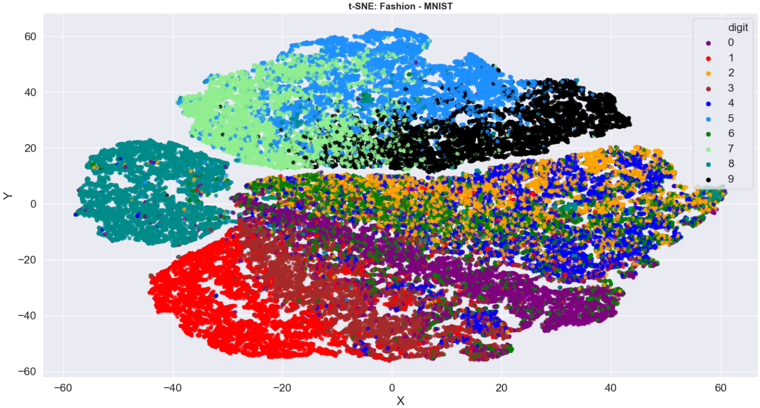
Our program - Fashion-MNIST



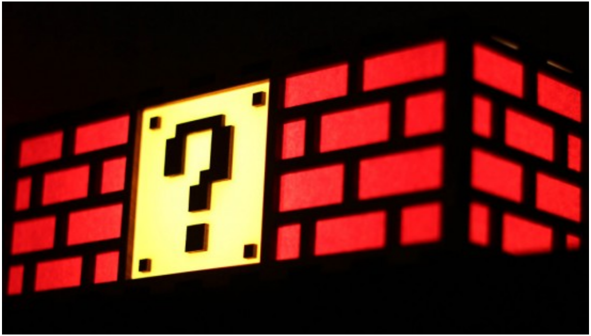
Our program - Fashion-MNIST(Alternative)



Our program - Fashion-MNIST changing the number of PCAs - decreased



Questions?



THE END

Thank you for your attention!

Sources

1. MNIST_784 photo
2. MNIST Fashion photo
3. Normal vs t-distribution photo
4. Question mark photo
5. https://www.datacamp.com/community/tutorials/introduction-t-sne
6. https://www.countbayesie.com/blog/2017/5/9/kullback-leibler-divergence-explained
7. https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding
8. StatQuest: t-SNE, Clearly Explained