t-SNE

t-Distributed Stochastic Neighbor Embedding

CAMBRIDGE SPARK

Overview

1. Motivation for using t-SNE
2. Understand the intuition behind t-SNE
3. Delve into the mathematics of how the model fits
4. Consider the drawbacks

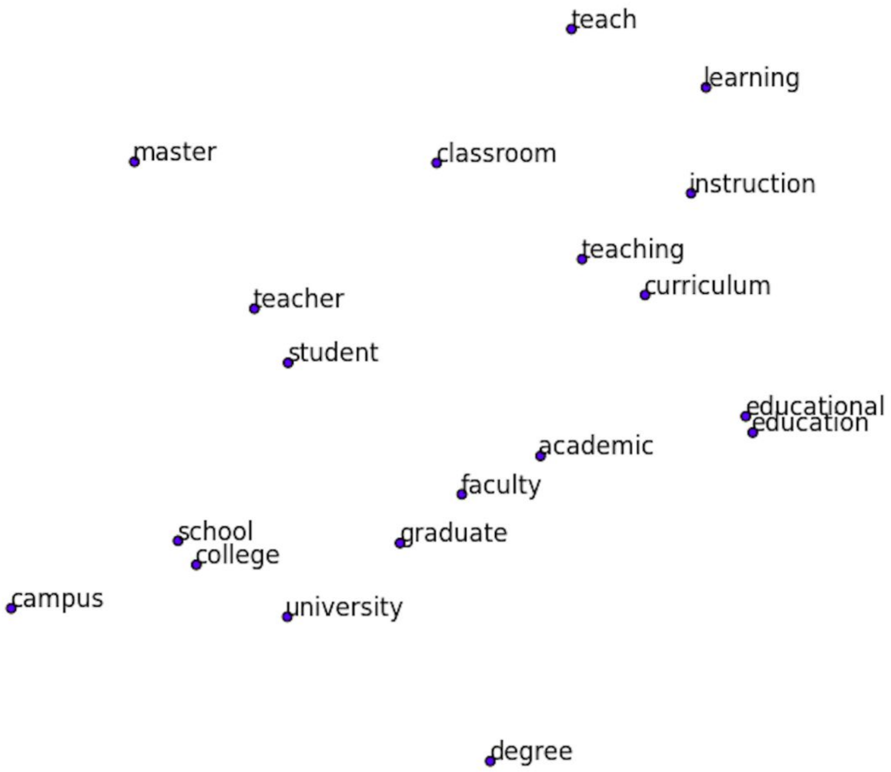
Why t-SNE?

CAMBRIDGE SPARK

Why t-SNE?

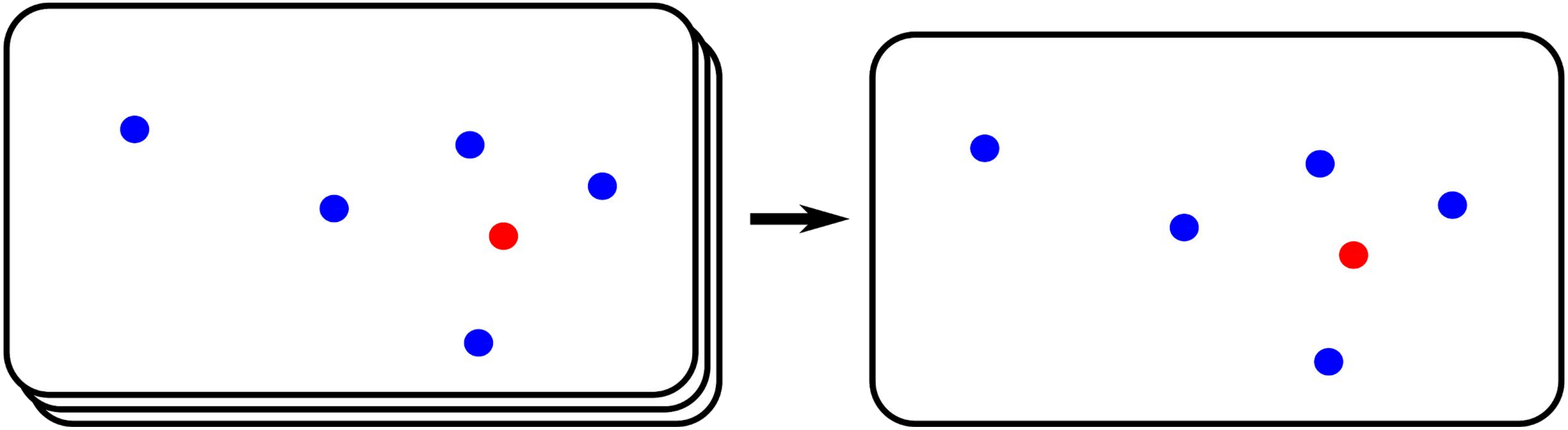
Why t-SNE?

Why t-SNE?



CAMBRIDGE SPARK

Visualising High Dimensional Data



CAMBRIDGE SPARK

Similarities as Probability

1. t-SNE treats distances in the original space as **probabilities**

CAMBRIDGE SPARK

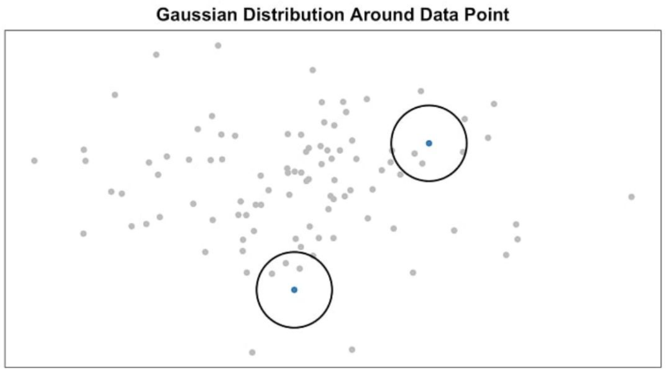
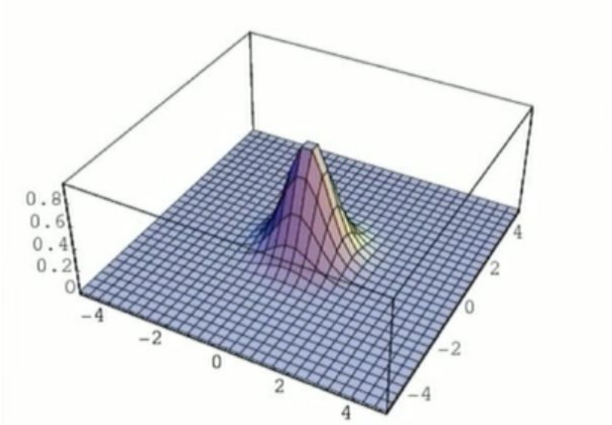
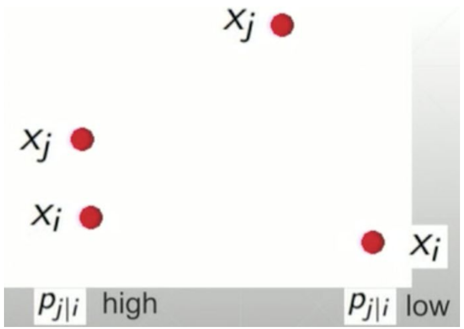


Figure 2—Measuring pairwise similarities in the high-dimensional space

SNE converts the pairwise Euclidean distances

between points into a probability density





Similarities as Probability

1. t-SNE treats distances in the original space as **probabilities**

2. For each **pair** of points, compute the conditional probability:

$$p_{j|i} = \frac{\exp\left(-d(\boldsymbol{x}_i, \boldsymbol{x}_j)/(2\sigma_i^2)\right)}{\sum_{i \neq k} \exp\left(-d(\boldsymbol{x}_i, \boldsymbol{x}_k)/(2\sigma_i^2)\right)}, \quad p_{i|i} = 0$$

Similarities as Probability

1. t-SNE treats distances in the original space as **probabilities**

2. For each **pair** of points, compute the conditional probability:

$$p_{j|i} = \frac{\exp\left(-d(\boldsymbol{x}_i, \boldsymbol{x}_j)/(2\sigma_i^2)\right)}{\sum_{i \neq k} \exp\left(-d(\boldsymbol{x}_i, \boldsymbol{x}_k)/(2\sigma_i^2)\right)}, \quad p_{i|i} = 0$$

3. Which are used to generate the joint probabilities:

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

CAMBRIDGE SPARK

Mapping to Low Dimensions

Lay out points in the low dimensional space and compute the **probabilities:**

Mapping to Low Dimensions

Lay out points in the low dimensional space and compute the **probabilities:**

$$q_{ij} = \frac{(1+||\boldsymbol{y}_i - \boldsymbol{y}_j)||^2)^{-1}}{\sum_{k \neq l}(1+||\boldsymbol{y}_k - \boldsymbol{y}_l)||^2)^{-1}}$$

CAMBRIDGE SPARK

Extra

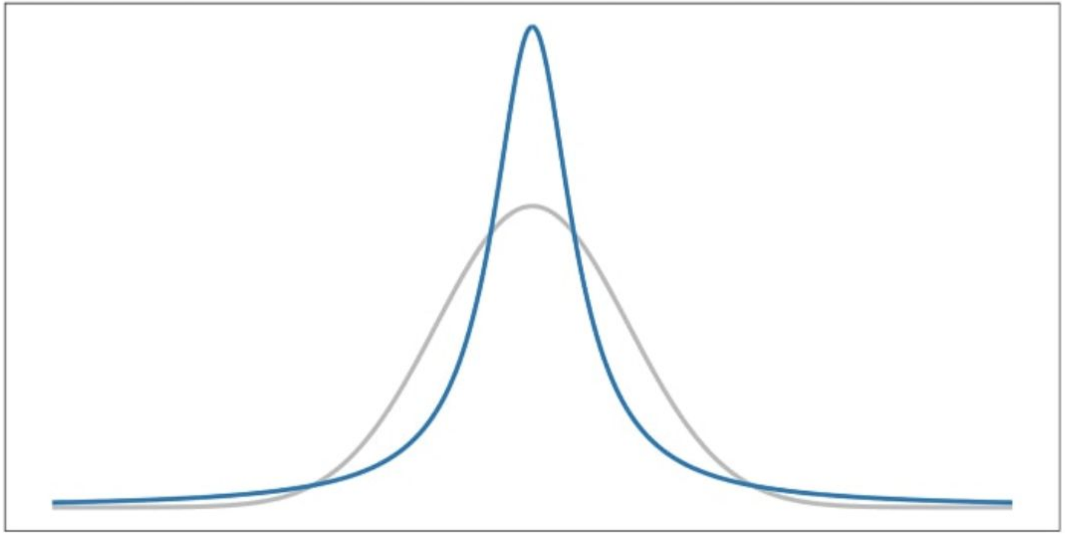


Figure 3—Normal vs Student t-distribution

CAMBRIDGE SPARK

Mapping to Low Dimensions

Lay out points in the low dimensional space and compute the **probabilities:**

$$q_{ij} = \frac{(1+||\boldsymbol{y}_i - \boldsymbol{y}_j)||^2)^{-1}}{\sum_{k \neq l}(1+||\boldsymbol{y}_k - \boldsymbol{y}_l)||^2)^{-1}}$$

Before calculating and *minimizing* the difference between q_{ij} and p_{ij}:

$$KL(P|Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

CAMBRIDGE SPARK

Summary

- t-SNE, unlike PCA, is not a linear projection. It uses the **local relationships** between points to create a low-dimensional mapping. This allows it to capture **non-linear structure**.
- t-SNE creates a **probability distribution** using the **Gaussian** distribution that defines the relationships between the points in high-dimensional space.
- t-SNE uses the **Student t-distribution** to **recreate** the probability distribution in low-dimensional space. This prevents the **crowding problem**, where points tend to get crowded in low-dimensional space due to the **curse of dimensionality**.
- t-SNE optimizes the embeddings directly using gradient descent. The cost function is **non-convex** though, meaning there is the risk of getting stuck in local minima. t-SNE uses multiple tricks to try to avoid this problem.

Hands-on session

t-sne.ipynb

TSNE

```
from sklearn.manifold import TSNE
digits_tsne = TSNE(
    n_components=2,
    perplexity=40,
    verbose=2).fit_transform(digits_50)
```

Extra

```
sklearn.manifold.TSNE(
        n_components=2,
        perplexity=30.0,
        early_exaggeration=4.0,
        learning_rate=1000.0,
        n_iter=1000,
        n_iter_without_progress=30,
        min_grad_norm=1e-07,
        metric='euclidean',
        init='random',
        verbose=0,
        random_state=None,
        method='barnes_hut',
        angle=0.5)
```

Perplexity is a global parameter denoting the effective number of neighbors.
(Recommended range: 5-50)

Extra

Tips from scikit-learn:

- “It is highly recommended to use another dimensionality reduction method (e.g. PCA for dense data or TruncatedSVD for sparse data) to reduce the number of dimensions to a reasonable amount (e.g. 50) if the number of features is very high. This will suppress some noise and speed up the computation of pairwise distances between samples.

- “The perplexity is related to the number of nearest neighbors that is used in other manifold learning algorithms. Larger datasets usually require a larger perplexity. Consider selecting a value between 5 and 50. The choice is not extremely critical since t-SNE is quite insensitive to this parameter.”

Extra

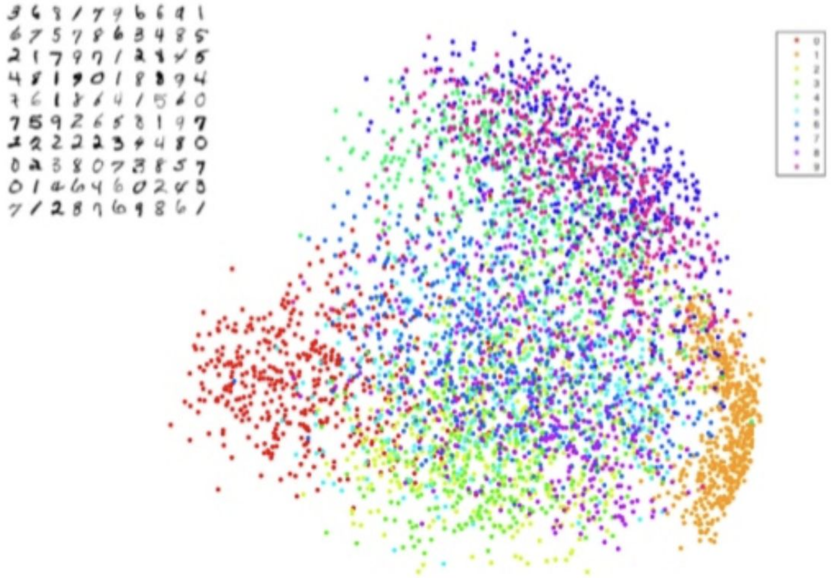
How should I set the perplexity in t-SNE?

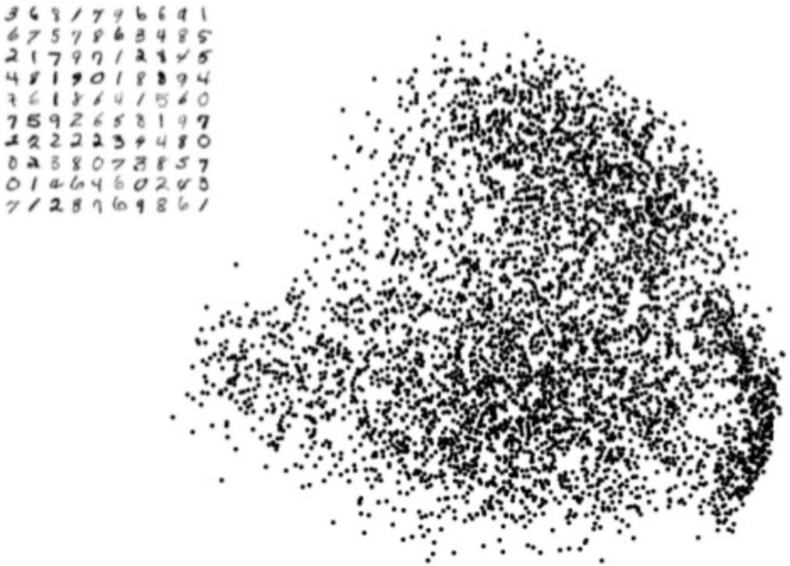
The performance of t-SNE is fairly robust under different settings of the perplexity. The most appropriate value depends on the density of your data. Loosely speaking, one could say that a larger / denser dataset requires a larger perplexity. Typical values for the perplexity range between 5 and 50.

What is perplexity anyway?

Perplexity is a measure for information that is defined as 2 to the power of the Shannon entropy. The perplexity of a fair die with k sides is equal to k. In t-SNE, the perplexity may be viewed as a knob that sets the number of effective nearest neighbors. It is comparable with the number of nearest neighbors k that is employed in many manifold learners.

Why not PCA?





CAMBRIDGE SPARK

Useful links

http://www.cs.toronto.edu/~hinton/absps/tsne.pdf

https://www.youtube.com/watch?v=EMD106bB2vY

https://distill.pub/2016/misread-tsne/

http://mlexplained.com/2018/09/14/paper-dissected-visualizing-data-using-t-sne-explained/

https://www.youtube.com/watch?v=aStvaXMhGGs

https://www.kdnuggets.com/2018/08/introduction-t-sne-python.html

http://mlexplained.com/2018/09/14/paper-dissected-visualizing-data-using-t-sne-explained/

https://www.oreilly.com/learning/an-illustrated-introduction-to-the-t-sne-algorithm