CS184A/284A
AI in Biology and Medicine

Dimension Reduction and Data Visualization

Visualizing Data using t-SNE

Visualization and Dimensionality Reduction

Intuition behind t-SNE

Visualizing representations

Visualization is key to understand data easily

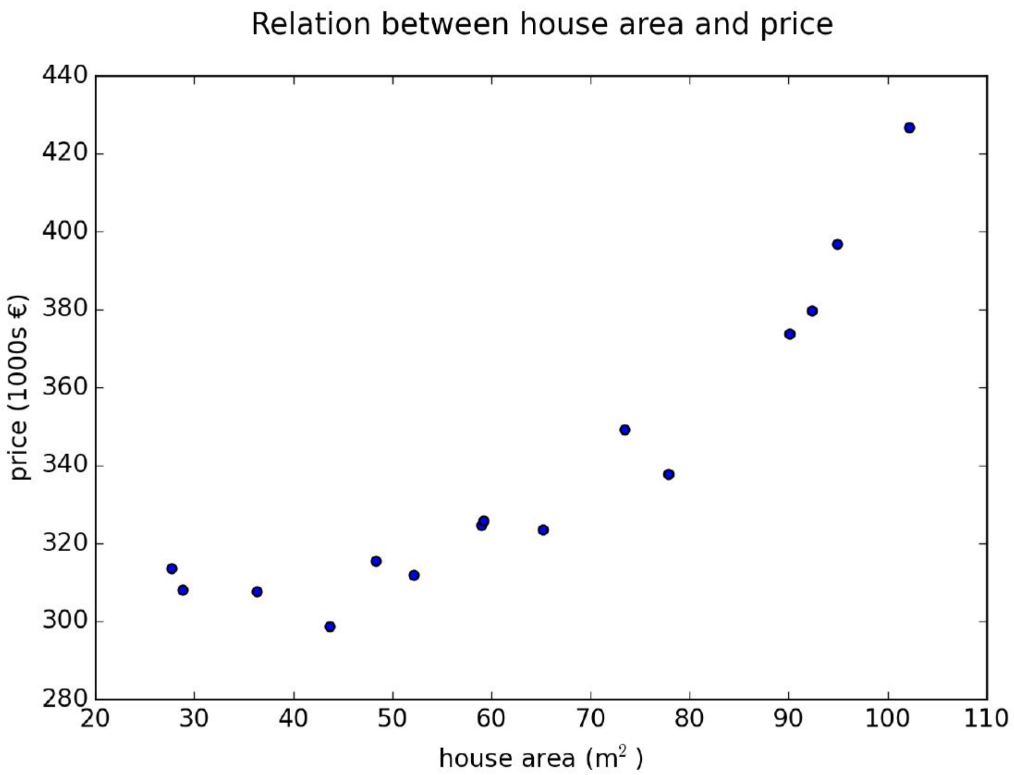
Data of house areas in m^2 and price in 1000s of euros.

Area	price
43.69	298.71
28.82	308.
102.22	426.68
36.32	307.53
48.35	315.4

Area	price
59.04	324.48
90.13	373.8
59.24	325.71
94.89	396.69
27.72	313.53

Area	price
65.2	323.43
92.38	379.56
77.86	337.77
73.48	349.15
52.19	311.86

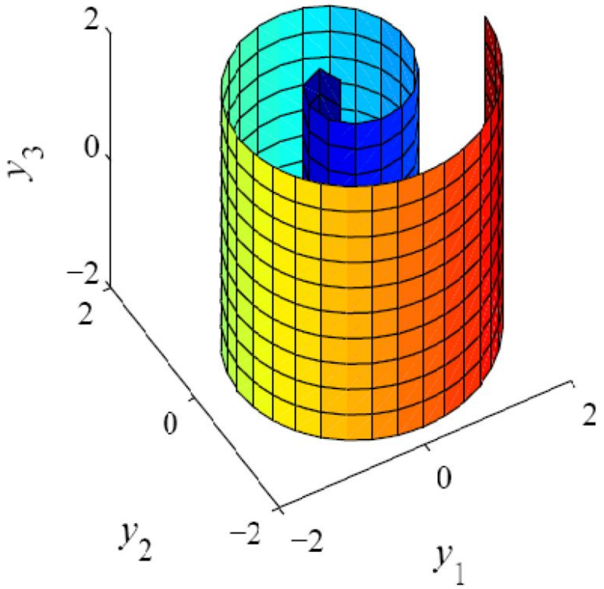
Question
Is the relation linear?



Relation between house area and price
440
420
400
380
360
340
320
300
280
price (1000s €)
20
30
40
50
60
70
80
90
100
110
house area (m2)

Dimensionality Reduction is a helpful tool for visualization

- Dimensionality reduction algorithms
  - Map high-dimensional data to a lower dimension
  - While preserving structure
- They are used for
  - Visualization
  - Performance
  - Curse of dimensionality
- A ton of algorithms exist
- t-SNE is specialised for visualization
- ... and has gained a lot of popularity



Dimensionality Reduction techniques solve optimization problems

$$\mathcal{X} = \{x_1, x_2, ..., x_n \in \mathbb{R}^h\} \rightarrow \mathcal{Y} = \{y_1, y_2, ..., y_n \in \mathbb{R}^l\}$$

$$\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$$

Three approaches for Dimensionality Reduction:

- Distance preservation
- Topology preservation
- Information preservation

t-SNE is distance-based but tends to preserve topology

SNE computes pair-wise similarities

SNE converts euclidean distances to similarities, that can be interpreted as probabilities.

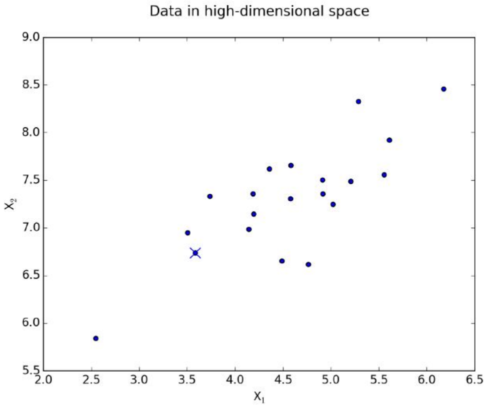
$$p_{j|i} = \frac{\exp(- \parallel x_i - x_j \parallel^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(- \parallel x_i - x_k \parallel^2 / 2\sigma_i^2)}$$

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

$$p_{i|i} = 0, q_{i|i} = 0$$

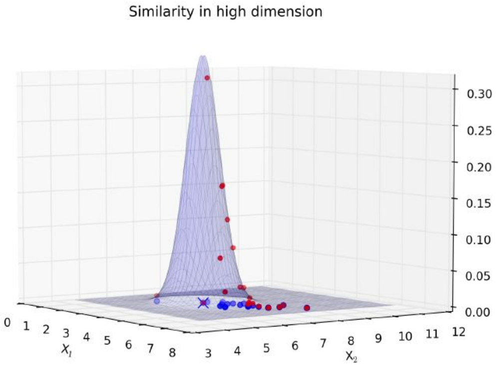
Hence the name *Stochastic Neighbor Embedding...*

Pair-wise similarities should stay the same

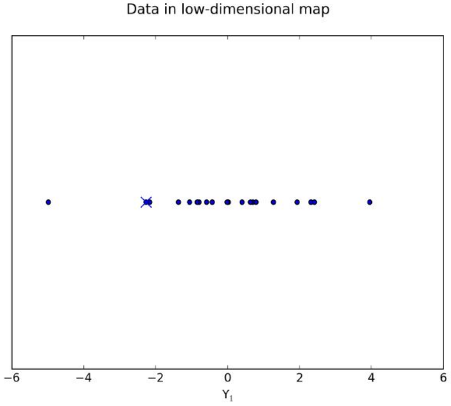


$p_{j|i}$

$\Leftrightarrow$

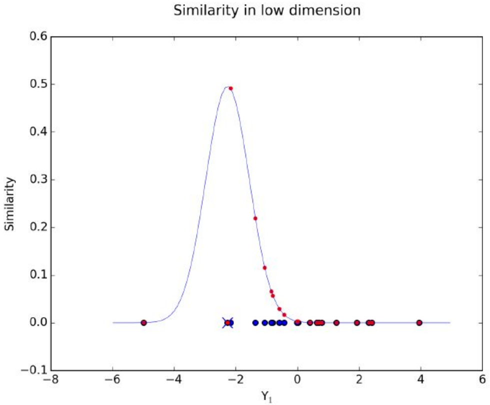


$\Downarrow$



$q_{j|i}$

$\Leftrightarrow$



Kullback-Leiber Divergence measures the faithfulness with which $q_{j|i}$ models $p_{j|i}$

- $P_i = \{p_{1|i}, p_{2|i}, ..., p_{n|i}\}$ and $Q_i = \{q_{1|i}, q_{2|i}, ..., q_{n|i}\}$ are the distributions on the neighbors of datapoint i.
- Kullback-Leiber Divergence (KL) compares two distributions.

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

- KL divergence is asymmetric
- KL divergence is always positive.
- We have our minimization problem: $\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$

Some remaining questions

$$p_{j|i} = \frac{\exp(-\parallel x_i - x_j \parallel^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\parallel x_i - x_k \parallel^2 / 2\sigma_i^2)}, \quad q_{j|i} = \frac{\exp(-\parallel y_i - y_j \parallel^2)}{\sum_{k \neq i} \exp(-\parallel y_i - y_k \parallel^2)}$$

Why radial basis function (exponential)?

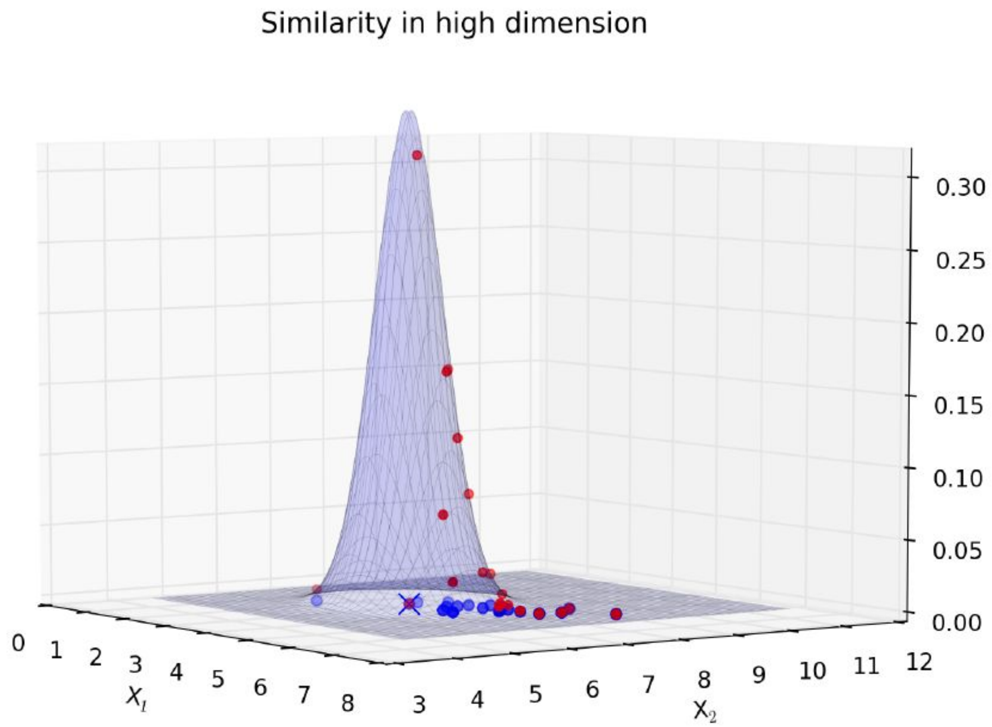
2. Why probabilities?

3. How do you choose i?

Why radial basis function (exponential)?

Focus on local geometry.

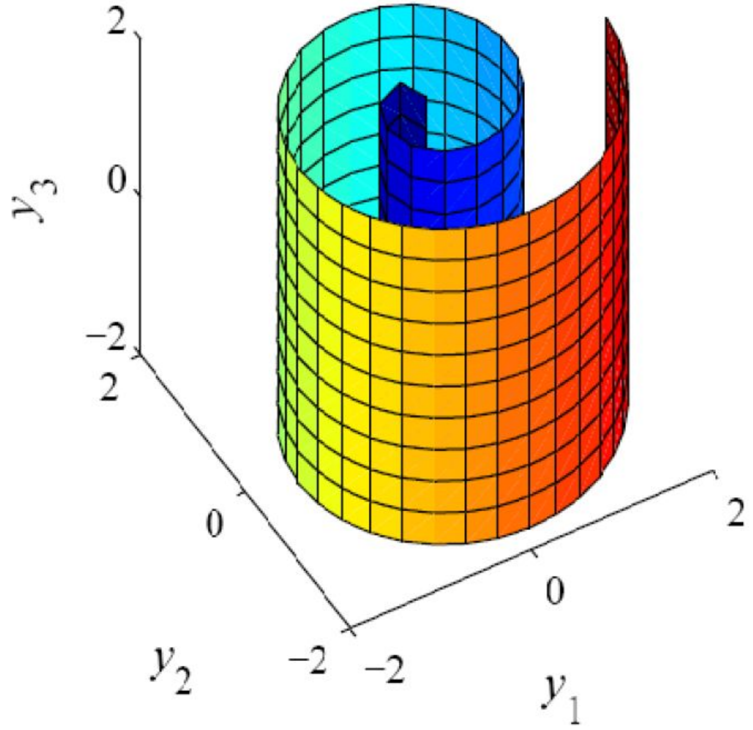
This is why t-SNE can be interpreted as topology-based



Why probabilities?

Small distance does not mean proximity on manifold.

Probabilities are appropriate to model this uncertainty

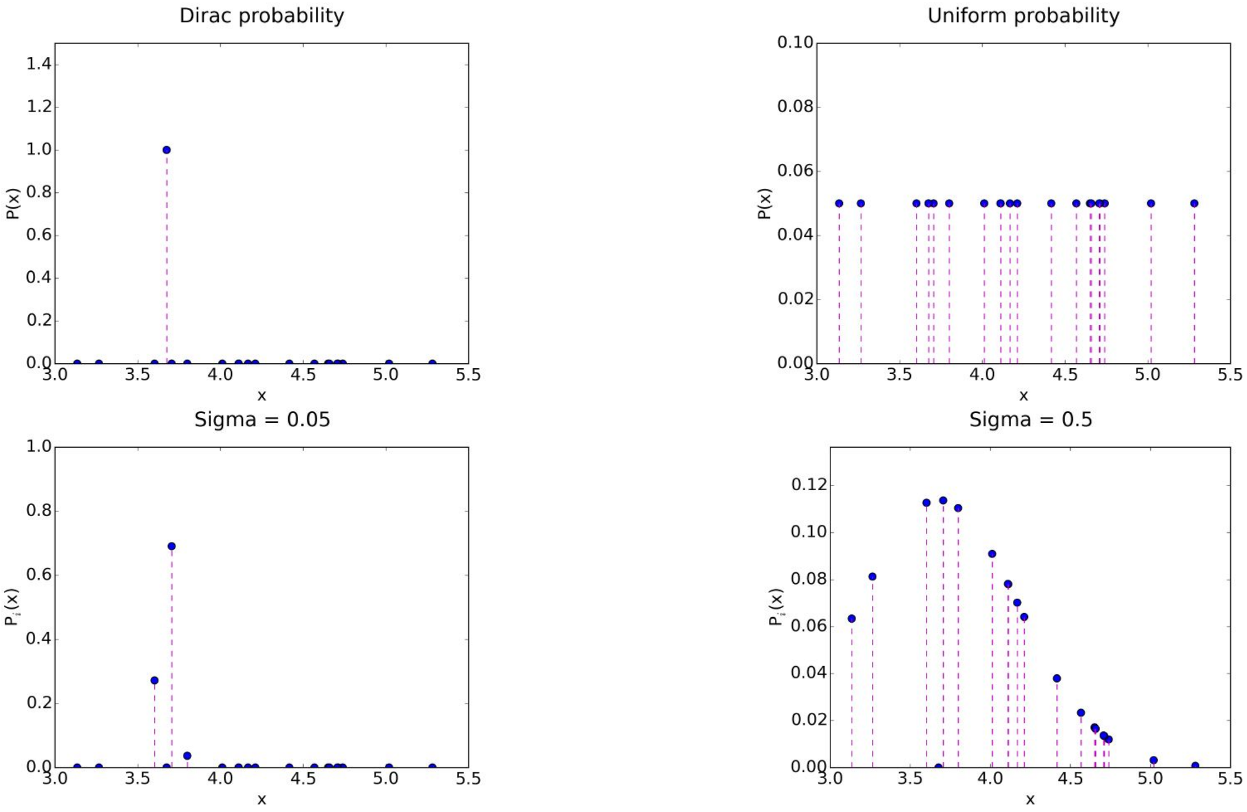


How do you choose σ_i?

The entropy of p increases with σ_i

Entropy

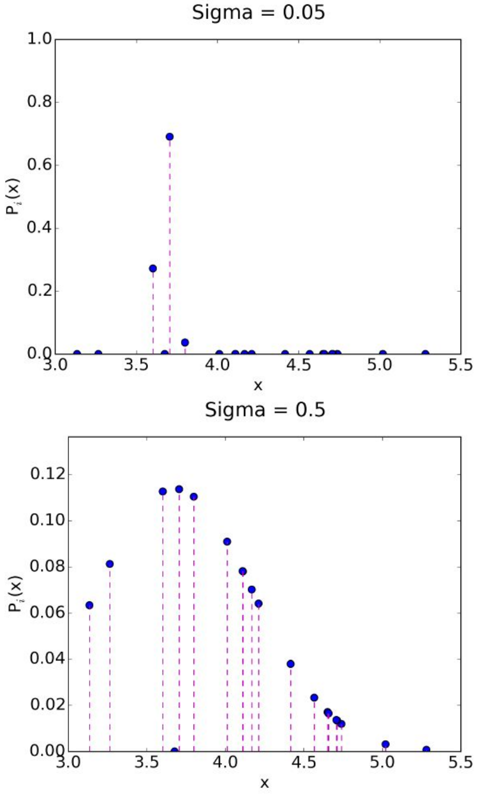
$H(p) = -\Sigma_i \, p_i \log_2 p_i$



Perplexity, a smooth measure of the # of neighbors.

Perplexity

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



$\Rightarrow$ Entropy of 1.055
Perplexity of 2.078

$\Rightarrow$ Entropy of 3.800
Perplexity of 13.929

From SNE to t-SNE.

SNE ⇒ Symmetric SNE ⇒ t-SNE

SNE

Modelisation:

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

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

Cost Function:

$$C = \sum_i KL(P_i||Q_i)$$

Derivatives:

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Symmetric SNE

Modelisation:

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

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

Cost Function:

$$C = KL(P||Q)$$

Derivatives:

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

- **Faster Computation**

t-SNE

Modelisation:

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

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

Cost Function:

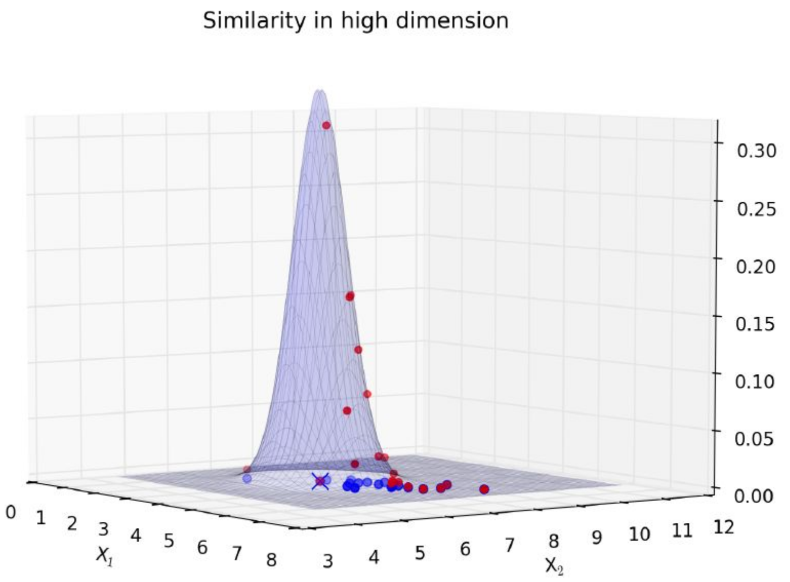
$$C = KL(P||Q)$$

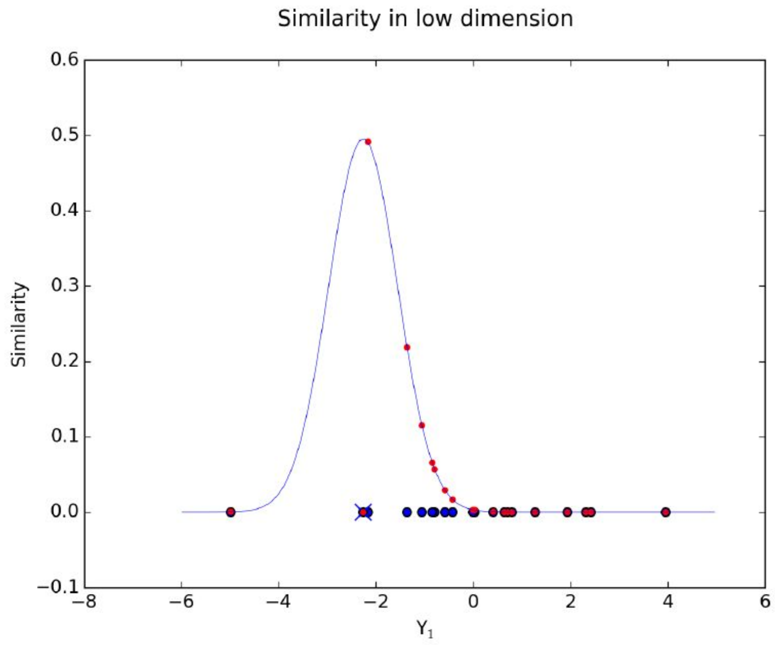
Derivatives:

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1+\| y_i - y_j \|^2)^{-1}$$

- **Even Faster Computation**
- **Better Behaviour**

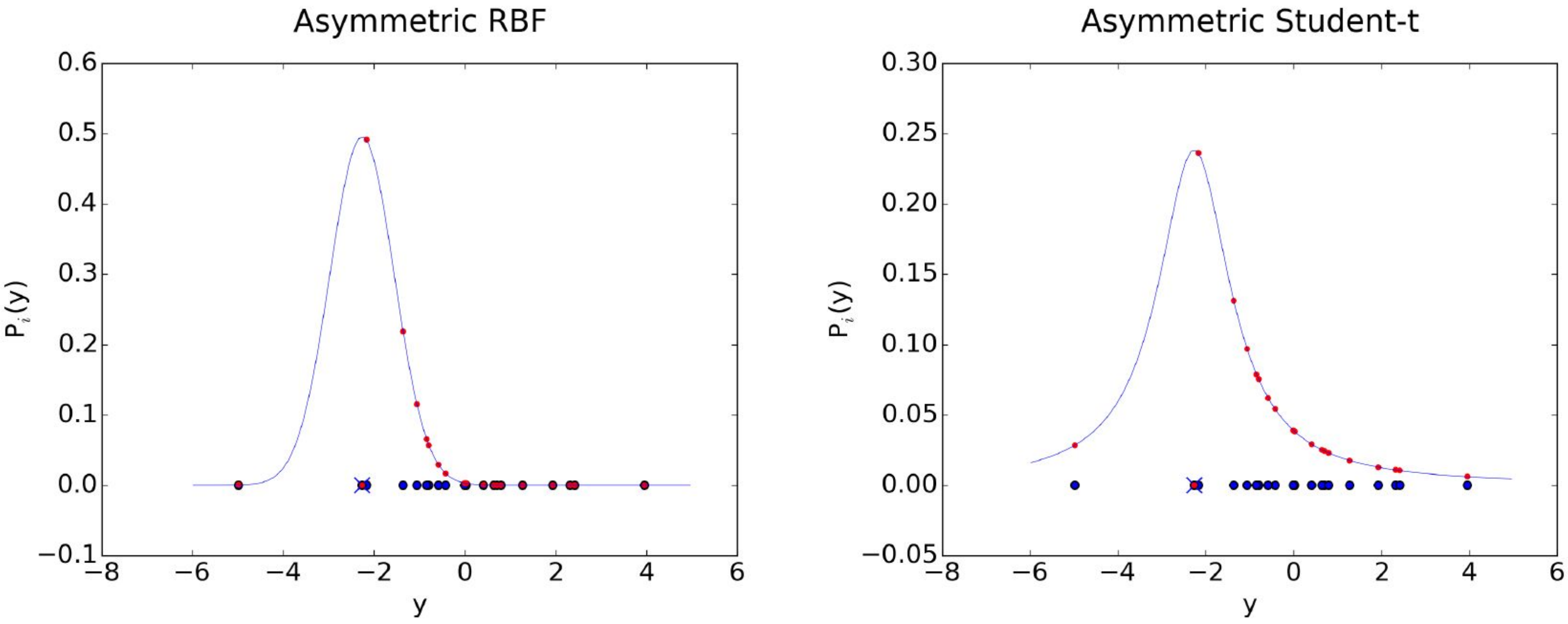
The "Crowding problem"





There is much more space in high dimensions.

Mismatched Tails can Compensate for Mismatched Dimensionalities



Student-t distribution has heavier tails.

Last but not least: Optimization

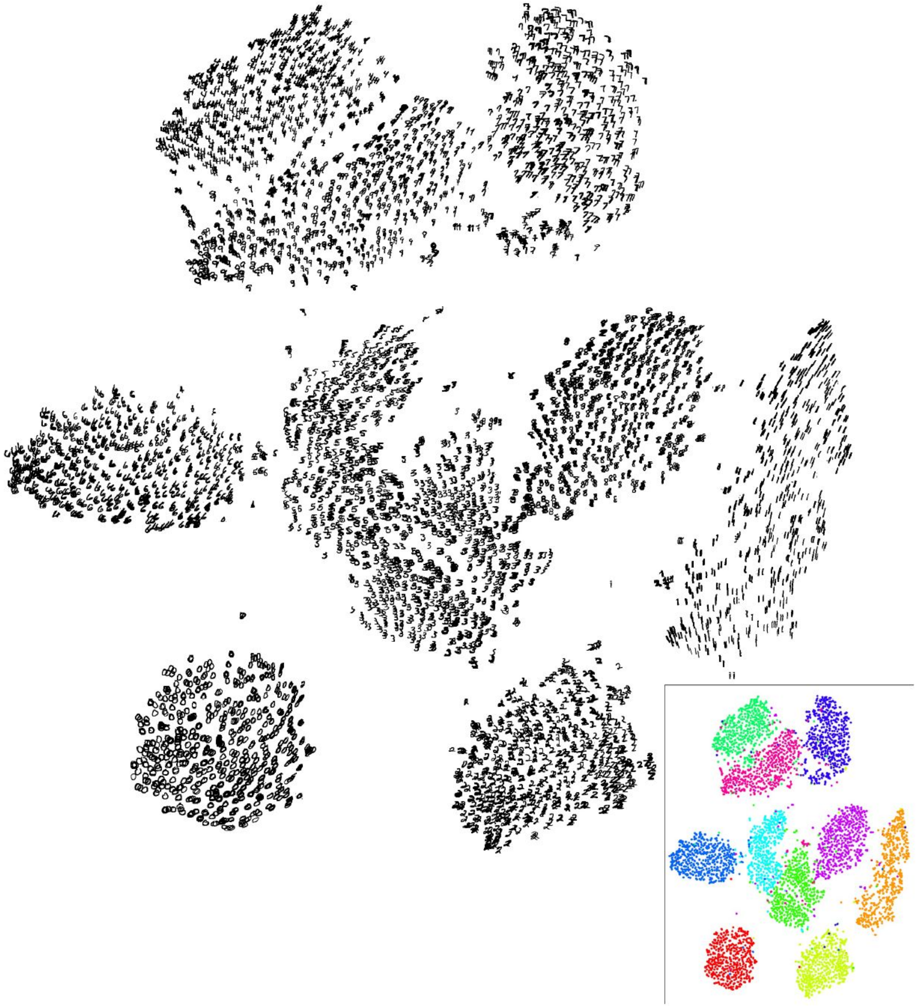
$$\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$$

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

- Non-convex
- Gradient descent + Momentum + Adaptive learning rate

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta(t)\frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)})$$

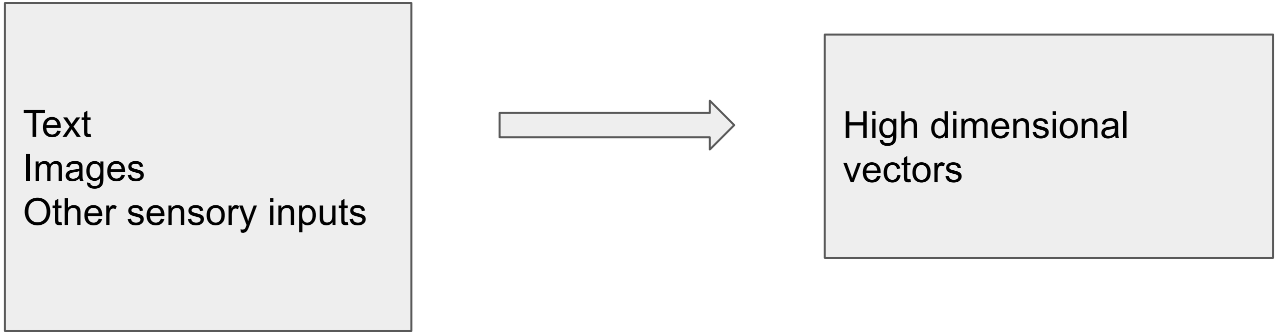
- Two tricks:
  - Early Compression
  - Early Exaggeration
- Illustration Colah's blog



Visualizing representations

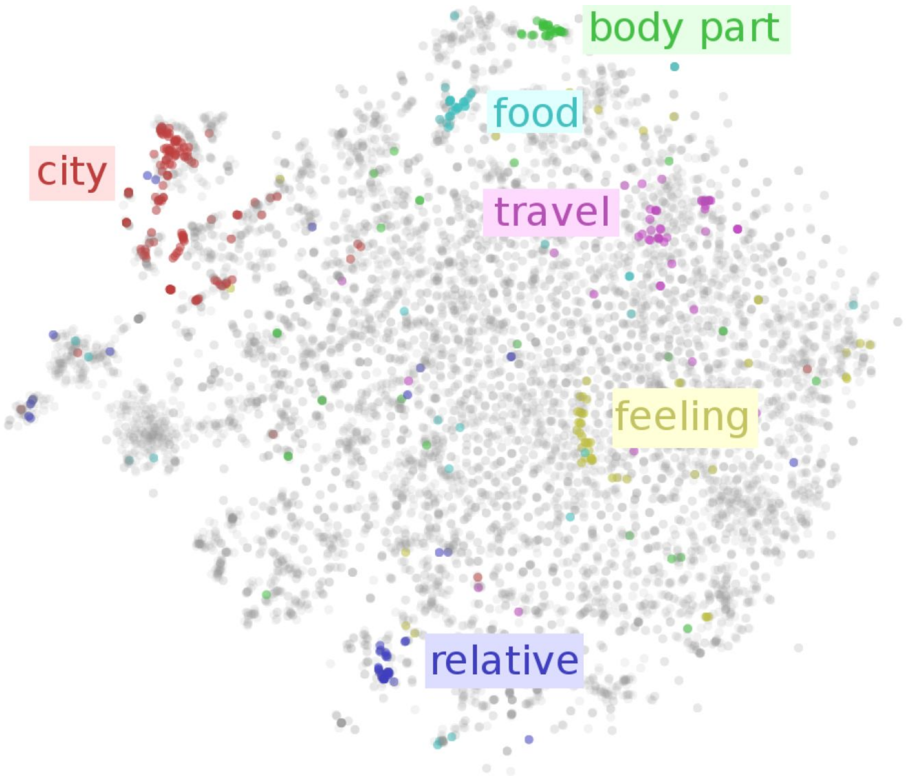
Mapping raw data to distributed representations

- Feature engineering is often laborious.
- New tendency is to automatically learn adequate features or representations.
- Ultimate goal: enable AI to extract useful features from raw sensory data.



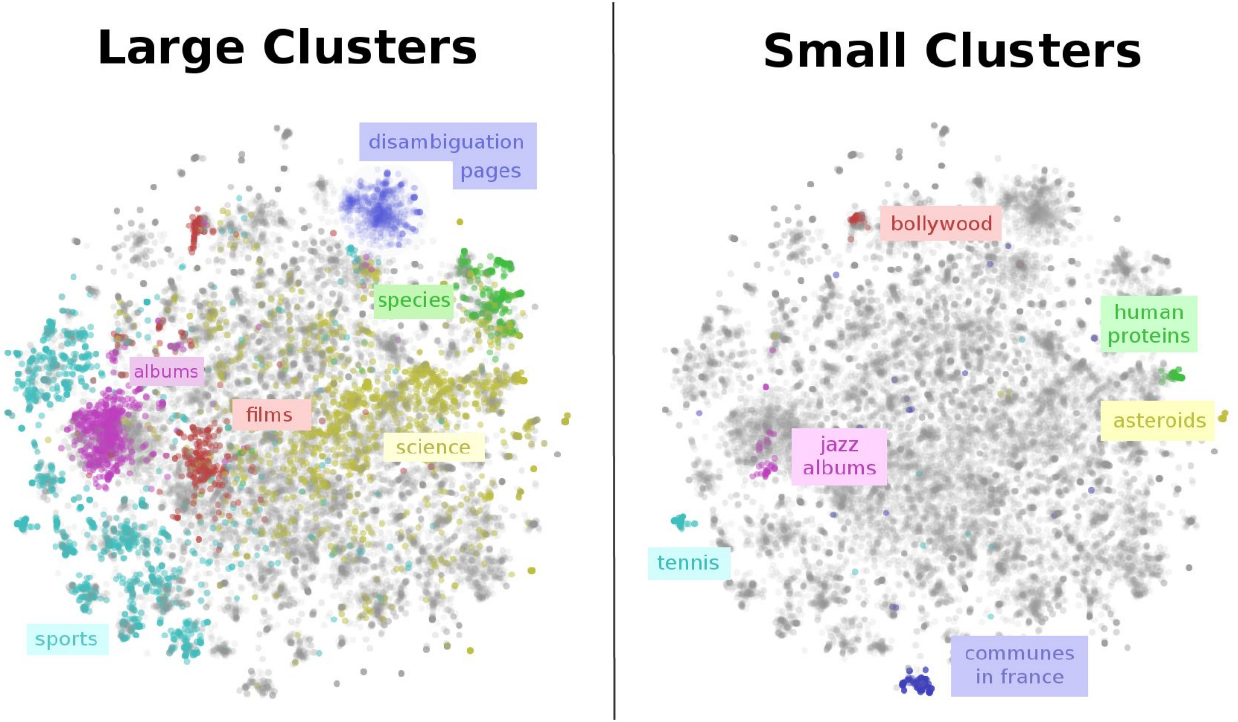
- t-SNE can be used to make sense of the learned representations!

Using t-SNE to explore a Word embedding



http://colah.github.io/

Explore a Wikipedia article embedding.

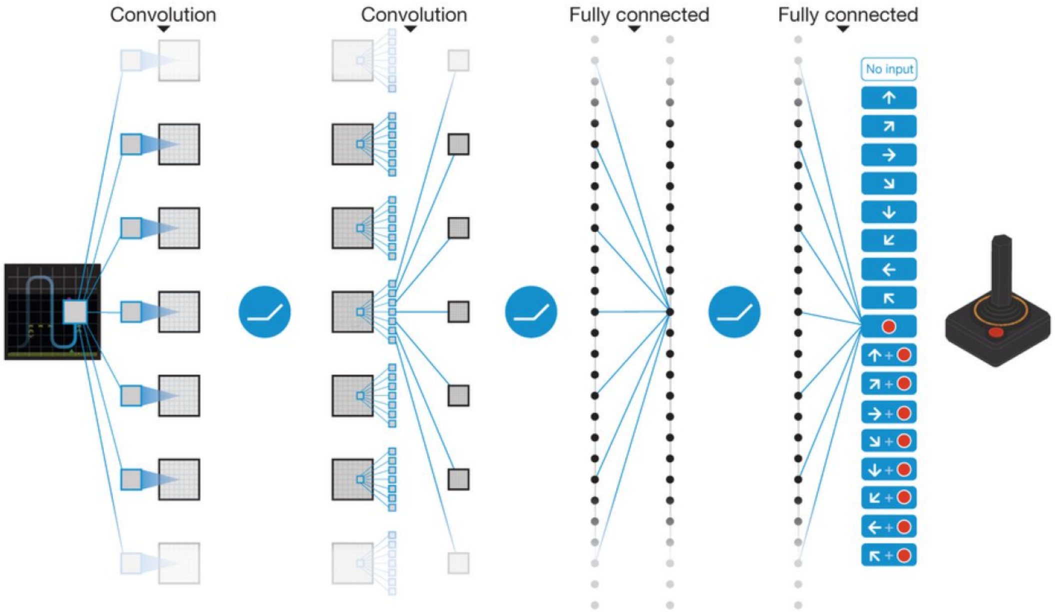


http://colah.github.io/

Exploring game state representations.

Google Deepmind plays Atari games.

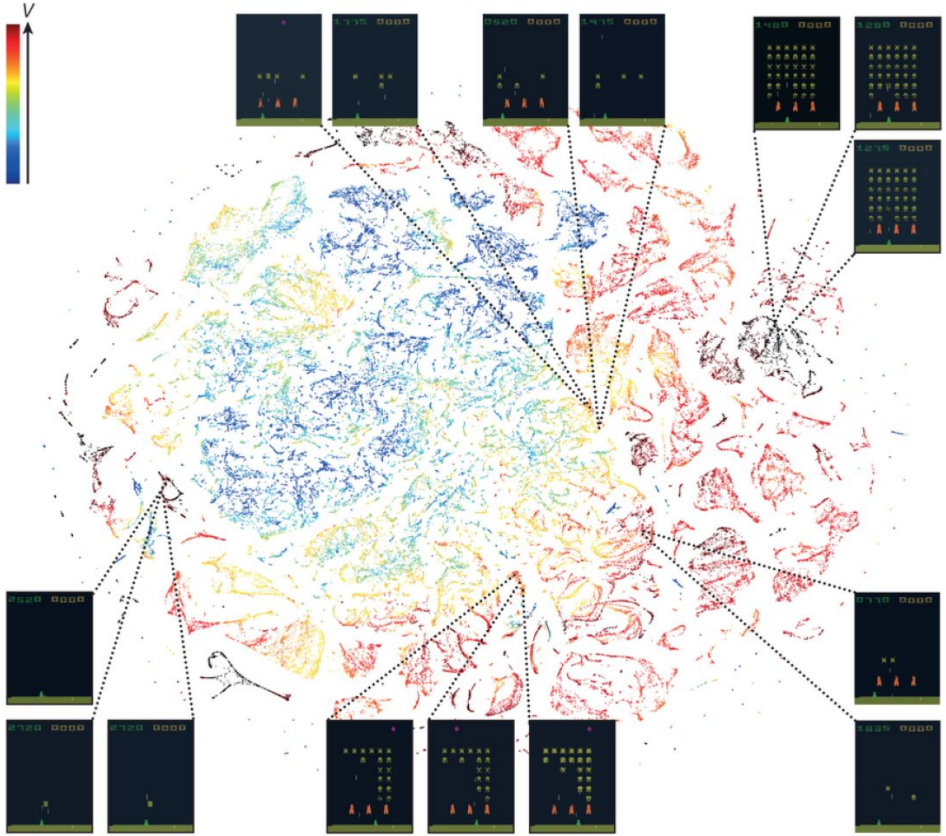
- A representation is learned with a convolutional neural network
- From 84x84x4 = 28.224 pixel values to 512 neurons.
- Predicts expected score if a certain action is taken.



Human-level control through deep reinforcement learning, V. Mnih et Al. (Nature,2015)

Exploring game state representations.

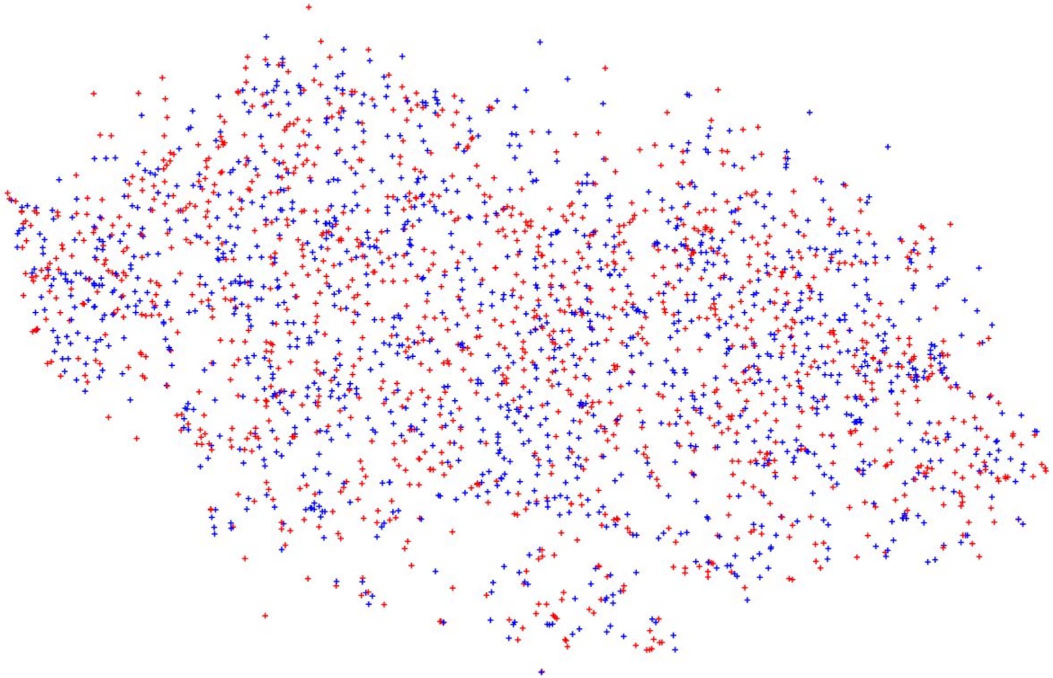
Google Deepmind plays
Atari games.



Human-level control through deep reinforcement learning, V. Mnih et Al. (Nature,2015)

Using t-SNE to explore image representations.

Classifying dogs and cats.

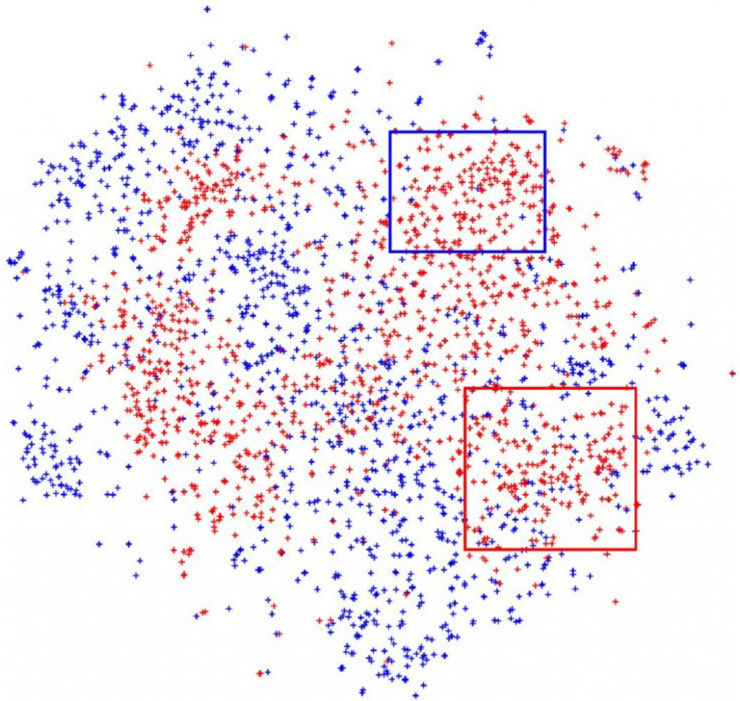


https://indico.io/blog/visualizing-with-t-sne/

Each data point is an image of a dog or a cat
red = cats, blue = dogs

Using t-SNE to explore image representations.

Classifying dogs and cats.

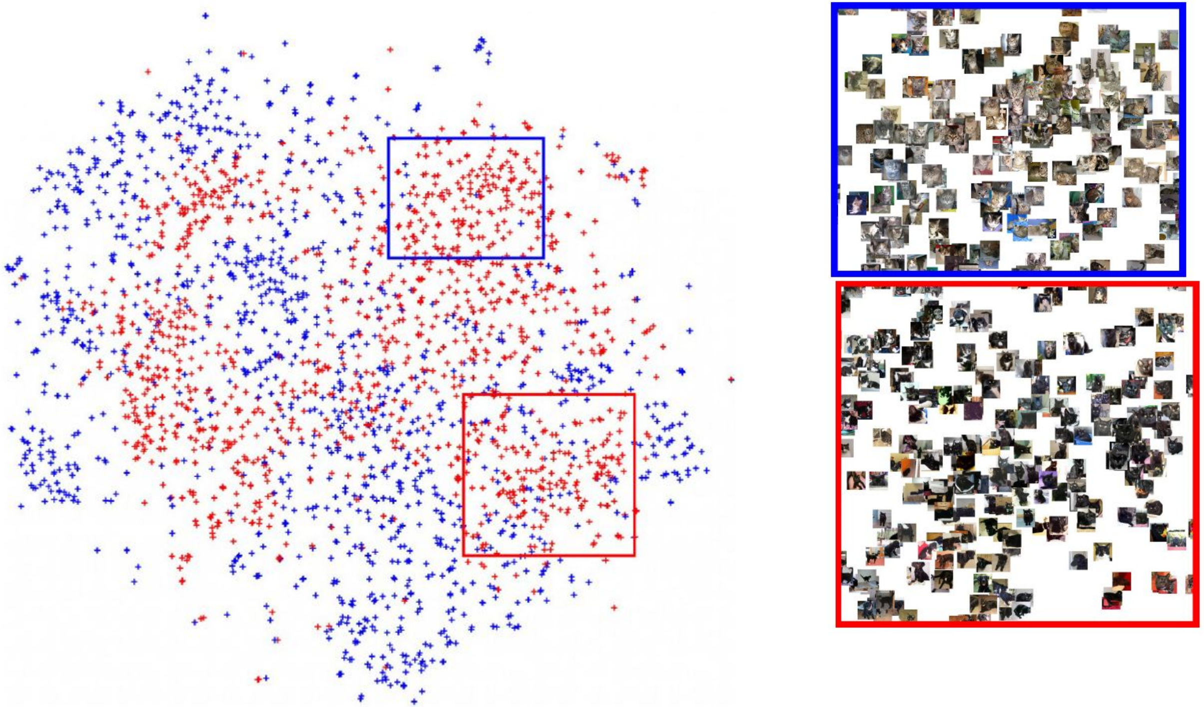


Representation
Convolutional net trained for Image Classification (1000 classes)

https://indico.io/blog/visualizing-with-t-sne/

Using t-SNE to explore image representations.

Classifying dogs and cats.



Representation
Convolutional net trained for Image Classification (1000 classes)

https://indico.io/blog/visualizing-with-t-sne/

Conclusion

- The t-SNE algorithm reduces dimensionality while preserving local similarity.
- The t-SNE algorithm has been build heuristically.
- t-SNE is commonly used to visualize representations.

Acknowledgement

Simon Carbonnelle for slides