The Zen of PCA, t-SNE, and Autoencoders
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space
• Linear reduction of dimensionality
  – Principle Component Analysis
  – Singular Value Decomposition
• Non-linear dimensionality reduction: embeddings
  – t-distributed Stochastic Network Embedding (t-SNE)
  – Building intuition: Playing with t-SNE parameters
• Deep Learning embeddings
  – Autoencoders
1. The biology: RNA-seq data
RNA-Seq characterizes RNA molecules

A Gene in genome

B pre-mRNA or ncRNA

C transcription

A mRNA

B splicing

C export to cytoplasm

High-throughput sequencing of RNAs at various stages of processing
RNA-Seq: De novo tx reconstruction / quantification

**Microarray technology**
- Synthesize DNA probe array, complementary hybridization
- Variations:
  - One long probe per gene
  - Many short probes per gene
  - Tiled k-mers across genome
- Advantage:
  - Can focus on small regions, even if few molecules / cell

**RNA-Seq technology:**
- Sequence short reads from mRNA, map to genome
- Variations:
  - Count reads mapping to each known gene
  - Reconstruct transcriptome de novo in each experiment
- Advantage:
  - Digital measurements, de novo
Expression Analysis Data Matrix

- Measure 20,000 genes in 100s of conditions

Each experiment measures expression of thousands of ‘spots’, typically genes

- Study resulting matrix
Clustering vs. Classification

Goal of Clustering: Group similar items that likely come from the same category, and in doing so reveal hidden structure

Goal of Classification: Extract features from the data that best assign new elements to ≥1 of well-defined classes

- Unsupervised learning
- Supervised learning
Clustering vs Classification

- **Objects** characterized by one or more features

- **Classification (supervised learning)**
  - Have labels for some points
  - Want a “rule” that will accurately assign labels to new points
  - Sub-problem: Feature selection
  - Metric: Classification accuracy

- **Clustering (unsupervised learning)**
  - No labels
  - Group points into clusters based on how “near” they are to one another
  - Identify structure in data
  - Metric: independent validation features
Today: Gene Expression, PCA, t-SNE, autoencoders

- Gene expression analysis: The Biology of RNA-seq
- Supervised (Classification) vs. unsupervised (Clustering)
  - Supervised: Differential expression analysis
  - Unsupervised: Embedding into lower dimensional space
  - Linear reduction of dimensionality
    - Principle Component Analysis
    - Singular Value Decomposition
  - Non-linear dimensionality reduction: embeddings
    - t-distributed Stochastic Network Embedding (t-SNE)
    - Building intuition: Playing with t-SNE parameters
- Deep Learning embeddings
  - Autoencoders
2. Supervised learning: differential gene expression
Statistical tests: example

- The alternative hypothesis $H_1$ is more expressive in terms of explaining the observed data.

null hypothesis

alternative hypothesis

- We need to find a way of testing whether this difference is significant.
Degrees of freedom

- How many degrees of freedom do we have in the two models?

\[ H_0 : \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N \left( \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \right) \]

\[ H_1 : \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N \left( \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right) \]
Degrees of freedom

- How many degrees of freedom do we have in the two models?

\[ H_0 : \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N \left( \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \right) \]

\[ H_1 : \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N \left( \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right) \]

- The observed data overwhelmingly supports \( H_1 \)
Test statistic

- Likelihood ratio statistic

\[
T(X^{(1)}, \ldots, X^{(n)}) = 2 \log \frac{P(X^{(1)}, \ldots, X^{(n)}|\hat{H}_1)}{P(X^{(1)}, \ldots, X^{(n)}|\hat{H}_0)}
\]  

(1)

Larger values of \( T \) imply that the model corresponding to the null hypothesis \( H_0 \) is much less able to account for the observed data.

- To evaluate the P-value, we also need to know the sampling distribution for the test statistic.

In other words, we need to know how the test statistic \( T(X^{(1)}, \ldots, X^{(n)}) \) varies if the null hypothesis \( H_0 \) is correct.
Test statistic cont’d

- For the likelihood ratio statistic, the sampling distribution is $\chi^2$ with degrees of freedom equal to the difference in the number of free parameters in the two hypotheses.

- Once we know the sampling distribution, we can compute the P-value

$$p = Prob\left( T(X^{(1)}, \ldots, X^{(n)}) \geq T_{obs} \mid H_0 \right)$$ (2)
What is the right distribution for modeling read counts?

\[ \lambda = \frac{\sum_{i=1}^{n} x_i}{n} \]

\[ f(x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!} \]

Poission?
Read count data is overdispersed for a Poisson
Use a Negative Binomial instead

\[
\sigma_{ij}^2 = \mu_{ij} + s_j^2 \nu_p(q_{ip(j)})
\]

Anders and Huber Genome Biology 2010, 11:R106
http://genomebiology.com/2010/11/10/R106
A Negative Binomial distribution is better (DESeq)

- \(i\) gene or isoform \(p\) condition
- \(j\) sample (experiment) \(p(j)\) condition of sample \(j\)
- \(m\) number of samples
- \(K_{ij}\) number of counts for isoform \(i\) in experiment \(j\)
- \(q_{ip}\) Average scaled expression for gene \(i\) condition \(p\)

\[
q_{ip} = \frac{1}{\text{# of replicates}} \sum_{j \text{ in replicates}} \frac{K_{ij}}{s_j}
\]

\[
\mu_{ij} = q_{ip(j)}s_j \quad \sigma_{ij}^2 = \mu_{ij} + s_j^2 v_p(q_{ip(j)})
\]

\(K_{ij} \sim NB(\mu_{ij}, \sigma_{ij}^2)\)
Figure 3 Testing for differential expression between conditions A and B: Scatter plot of log₂ ratio (fold change) versus mean. The red colour marks genes detected as differentially expressed at 10% false discovery rate when Benjamini-Hochberg multiple testing adjustment is used. The symbols at the upper and lower plot border indicate genes with very large or infinite log fold change. The corresponding volcano plot is shown in Supplementary Figure S8 in Additional file 2.
Hypergeometric test for gene set overlap significance

\[ P(k) = \frac{\binom{n_1}{k} \binom{N-n_1}{n_2-k}}{\binom{N}{n_2}} \]

\[ P(x \geq k) = \sum_{i=k}^{\min(n_1,n_2)} P(i) \]

N – total # of genes 1000
n1 - # of genes in set A 20
n2 - # of genes in set B 30
k - # of genes in both A and B 3

0.017 0.020
Bonferroni correction

• Total number of rejections of null hypothesis over all N tests denoted by R.

\[ Pr(R>0) \sim N\alpha \]

• Need to set \( \alpha' = Pr(R>0) \) to required significance level over all tests. Referred to as the experimentwise error rate.

• With 100 tests, to achieve overall experimentwise significance level of \( \alpha'=0.05 \):

\[ 0.05 = 100\alpha \]

\[ \Rightarrow \alpha = 0.0005 \]

• Pointwise significance level of 0.05%.
Example - Genome wide association screens

• Risch & Merikangas (1996).
• 100,000 genes.
• Observe 10 SNPs in each gene.
• 1 million tests of null hypothesis of no association.
• To achieve experimentwise significance level of 5%, require pointwise p-value less than $5 \times 10^{-8}$
Bonferroni correction - problems

• Assumes each test of the null hypothesis to be independent.
• If not true, Bonferroni correction to significance level is conservative.
• Loss of power to reject null hypothesis.
• Example: genome-wide association screen across linked SNPs – correlation between tests due to LD between loci.
Benjamini Hochberg

- Select False Discovery Rate $\alpha$
- Number of tests is $m$
- Sort $p$-values $P_{(k)}$ in ascending order (most significant first)
- Assumes tests are uncorrelated or positively correlated

1. For a given $\alpha$, find the largest $k$ such that $P_{(k)} \leq \frac{k}{m} \alpha$.

2. Reject the null hypothesis (i.e., declare discoveries) for all $H_{(i)}$ for $i = 1, \ldots, k$. 
Today: Gene Expression, PCA, t-SNE, autoencoders

- Gene expression analysis: The Biology of RNA-seq
- Supervised (Classification) vs. unsupervised (Clustering)
- Supervised: Differential expression analysis
  - Unsupervised: Embedding into lower dimensional space
    - Linear reduction of dimensionality
      - Principle Component Analysis
      - Singular Value Decomposition
    - Non-linear dimensionality reduction: embeddings
      - t-distributed Stochastic Network Embedding (t-SNE)
      - Building intuition: Playing with t-SNE parameters
- Deep Learning embeddings
  - Autoencoders
3. Unsupervised learning: dimensionality reduction
Dimensionality reduction has multiple applications

• Uses:
  – Data Visualization
  – Data Reduction
  – Data Classification
  – Trend Analysis
  – Factor Analysis
  – Noise Reduction

• Examples:
  – How many unique “sub-sets” are in the sample?
  – How are they similar / different?
  – What are the underlying factors that influence the samples?
  – Which time / temporal trends are (anti)correlated?
  – Which measurements are needed to differentiate?
  – How to best present what is “interesting”?
  – Which “sub-set” does this new sample rightfully belong?
A manifold is a topological space that locally resembles Euclidean space near each point.

A manifold embedding is a structure preserving mapping of a high dimensional space into a manifold.

Manifold learning learns a lower dimensional space that enables a manifold embedding.
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space

• Linear reduction of dimensionality
  – Principle Component Analysis
  – Singular Value Decomposition

• Non-linear dimensionality reduction: embeddings
  – t-distributed Stochastic Network Embedding (t-SNE)
  – Building intuition: Playing with t-SNE parameters

• Deep Learning embeddings
  – Autoencoders
4. Principal Component Analysis
Example data

• Example: 53 Blood and urine measurements (wet chemistry) from 65 people (33 alcoholics, 32 non-alcoholics)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8.000</td>
<td>4.8200</td>
<td>14.1000</td>
<td>41.0000</td>
<td>85.0000</td>
<td>29.0000</td>
<td>34.0000</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>7.3000</td>
<td>5.0200</td>
<td>14.7000</td>
<td>43.0000</td>
<td>86.0000</td>
<td>29.0000</td>
<td>34.0000</td>
</tr>
<tr>
<td>A3</td>
<td>4.3000</td>
<td>4.4800</td>
<td>14.1000</td>
<td>41.0000</td>
<td>91.0000</td>
<td>32.0000</td>
<td>35.0000</td>
</tr>
<tr>
<td>A4</td>
<td>7.5000</td>
<td>4.4700</td>
<td>14.9000</td>
<td>45.0000</td>
<td>101.0000</td>
<td>33.0000</td>
<td>33.0000</td>
</tr>
<tr>
<td>A5</td>
<td>7.3000</td>
<td>5.5200</td>
<td>15.4000</td>
<td>46.0000</td>
<td>84.0000</td>
<td>28.0000</td>
<td>33.0000</td>
</tr>
<tr>
<td>A6</td>
<td>6.9000</td>
<td>4.8600</td>
<td>16.0000</td>
<td>47.0000</td>
<td>97.0000</td>
<td>33.0000</td>
<td>34.0000</td>
</tr>
<tr>
<td>A7</td>
<td>7.8000</td>
<td>4.6800</td>
<td>14.7000</td>
<td>43.0000</td>
<td>92.0000</td>
<td>31.0000</td>
<td>34.0000</td>
</tr>
<tr>
<td>A8</td>
<td>8.6000</td>
<td>4.8200</td>
<td>15.8000</td>
<td>42.0000</td>
<td>88.0000</td>
<td>33.0000</td>
<td>37.0000</td>
</tr>
<tr>
<td>A9</td>
<td>5.1000</td>
<td>4.7100</td>
<td>14.0000</td>
<td>43.0000</td>
<td>92.0000</td>
<td>30.0000</td>
<td>32.0000</td>
</tr>
</tbody>
</table>

• Trivariate plot
Principal Component = axis of greatest variability

Suppose we have a population measured on $p$ random variables $X_1, \ldots, X_p$. Note that these random variables represent the $p$-axes of the Cartesian coordinate system in which the population resides. Our goal is to develop a new set of $p$ axes (linear combinations of the original $p$ axes) in the directions of greatest variability:

This is accomplished by rotating the axes.
Data projected onto PC1

Figure 1A

Figure 1B
Selecting Principal Components

• Given m points in a n dimensional space, for large n, how does one project on to a 1 dimensional space?
• Formally, minimize sum of squares of distances to the line.

• Why sum of squares? Because it allows fast minimization, assuming the line passes through 0
**Linear Algebra Review**

- **Eigenvalues** *(for a square $m \times m$ matrix $S$)*

  \[ S\mathbf{v} = \lambda \mathbf{v} \]

  (right) eigenvector \hspace{2cm} eigenvalue

  \[ \mathbf{v} \in \mathbb{R}^m \neq 0 \hspace{1cm} \lambda \in \mathbb{R} \]

- **How many eigenvalues** are there at most?

  \[ S\mathbf{v} = \lambda \mathbf{v} \iff (S - \lambda \mathbf{I}) \mathbf{v} = 0 \]

  only has a non-zero solution if \[ |S - \lambda \mathbf{I}| = 0 \]

  this is a $m$-th order equation in $\lambda$ which can have **at most $m$ distinct solutions** (roots of the characteristic polynomial) - can be complex even though $S$ is real.

**Example**

\[
\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}
\]
Eigenvalues & Eigenvectors

- For symmetric matrices, eigenvectors for distinct eigenvalues are **orthogonal**

\[ S \mathbf{v}_{\{1,2\}} = \lambda_{\{1,2\}} \mathbf{v}_{\{1,2\}}, \text{ and } \lambda_1 \neq \lambda_2 \implies \mathbf{v}_1 \cdot \mathbf{v}_2 = 0 \]

- All eigenvalues of a real symmetric matrix are **real**.

For complex \( \lambda \), if \( |S - \lambda I| = 0 \) and \( S = S^T \) \( \implies \lambda \in \mathbb{R} \)

- All eigenvalues of a positive semidefinite matrix are **non-negative**

\[ \forall \mathbf{w} \in \mathbb{R}^n, \mathbf{w}^T S \mathbf{w} \geq 0, \text{ then if } S \mathbf{v} = \lambda \mathbf{v} \implies \lambda \geq 0 \]
Eigen/diagonal Decomposition

Let \( S \in \mathbb{R}^{m \times m} \) be a square matrix with \( m \) linearly independent eigenvectors (a “non-defective” matrix)

\[
S = U \Lambda U^{-1}
\]

- (cf. matrix diagonalization theorem)

Theorem: Exists an eigen decomposition

- Columns of \( U \) are eigenvectors of \( S \)
- Diagonal elements of \( \Lambda \) are eigenvalues of \( S \)

\[ \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_m), \quad \lambda_i \geq \lambda_{i+1} \]
Symmetric Eigen Decomposition

• If $S \in \mathbb{R}^{m \times m}$ is a symmetric matrix:

  • **Theorem**: Exists a (unique) eigen decomposition $S = QAQ^T$

• where $Q$ is orthogonal:
  – $Q^{-1} = Q^T$
  – Columns of $Q$ are normalized eigenvectors
  – Columns are orthogonal.
  – (everything is real)
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space
• Linear reduction of dimensionality
  – Principle Component Analysis
  – Singular Value Decomposition
• Non-linear dimensionality reduction: embeddings
  – t-distributed Stochastic Network Embedding (t-SNE)
  – Building intuition: Playing with t-SNE parameters
• Deep Learning embeddings
  – Autoencoders
5. Singular value decomposition (general $m \times n$ matrices)
Singular Value Decomposition

For an $m \times n$ matrix $A$ of rank $r$ there exists a factorization (Singular Value Decomposition = SVD) as follows:

$$A = U\Sigma V^T$$

where $U$ is $m \times m$, $V$ is $n \times n$, and $\Sigma$ is $m \times n$.

The columns of $U$ are orthogonal eigenvectors of $AA^T$.

The columns of $V$ are orthogonal eigenvectors of $A^TA$.

Eigenvalues $\lambda_1 \ldots \lambda_r$ of $AA^T$ are the eigenvalues of $A^TA$.

$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

Singular values.
Geometric interpretation of SVD

\[ M = U \cdot \Sigma \cdot V^* \]

\[ Mx = M(x) = U( S( V^*(x) ) ) \]
Singular Value Decomposition

- Illustration of SVD dimensions and sparseness.
Singular Value Decomposition-example

- Let

\[
A = \begin{bmatrix}
1 & -1 \\
0 & 1 \\
1 & 0
\end{bmatrix}
\]

Thus \( m=3, \ n=2 \). Its SVD is

\[
\begin{bmatrix}
0 & 2/\sqrt{6} & 1/\sqrt{3} \\
1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\
1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3}
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
0 & \sqrt{3} \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
1/\sqrt{2} & 1/\sqrt{2} \\
1/\sqrt{2} & -1/\sqrt{2}
\end{bmatrix}
\]

Typically, the singular values arranged in decreasing order.
Low-rank Approximation

• SVD can be used to compute optimal low-rank approximations.

• Approximation problem: Find $A_k$ of rank $k$ such that

$$A_k = \min_{X: \text{rank}(X) = k} \|A - X\|_F$$

*Frobenius norm (aka Euclidian norm)*

$$\|A\|_F \equiv \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}.$$  

$A_k$ and $X$ are both $m \times n$ matrices.
Typically, want $k < < r$. 
Low-rank Approximation

- **Solution via SVD**

\[ A_k = U \, \text{diag}(\sigma_1, \ldots, \underbrace{\sigma_k, 0, \ldots, 0}_r) V^T \]

set smallest \( r-k \) singular values to zero

\[ A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T \]

column notation: sum of rank 1 matrices

- **Error:**

\[ \min_{X: \text{rank}(X)=k} \| A - X \|_F = \| A - A_k \|_F = \sigma_{k+1} \]
Principle Component Analysis (PCA)

- How do we find the eigenvectors $v_i$?
- We use **singular value decomposition** to decompose $\Sigma$ into an orthogonal rotation matrix $U$ and a diagonal scaling matrix $S$:

  \[
  \Sigma = U S U^T \\
  \Sigma U = (USU^T)U \\
  = US
  \] (22)

  \[
  \Sigma U = (USU^T)U \\
  = US
  \] (23)

- The columns of $U$ are the $v_i$, and $S$ is the diagonal matrix of eigenvalues $\lambda_i^2$.
PCA of MNIST digits
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space
• Linear reduction of dimensionality
  – Principle Component Analysis
  – Singular Value Decomposition

• Non-linear dimensionality reduction: embeddings
  – t-distributed Stochastic Network Embedding (t-SNE)
    – Building intuition: Playing with t-SNE parameters
• Deep Learning embeddings
  – Autoencoders
6. Non-linear embeddings: t-SNE
Distance Preservation  
Neighbor Preservation

High Dim
Neighborhood not preserved
Neighborhood preserved
Measure pairwise distances in high dimensional space

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

Set the bandwidth \(\sigma_i\) such that the conditional has a fixed perplexity (effective number of neighbors) \(\text{Perp}(P_i) = 2^H(P_i)\), typical value is about 5 to 50

Shannon entropy of \(P_i\)
We want to choose an embedding that minimizes divergence between low and high dimension similarities.

- Similarity of datapoints in High Dimension
  \[ p_{ij} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)}{\sum_{k \neq l} \exp\left(-\frac{\|x_l - x_k\|^2}{2\sigma^2}\right)} \]

- Similarity of datapoints in Low Dimension
  \[ q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}} \]
Low dimensional embedding using a Student t-distribution to avoid overcrowding

Red – Student t-distribution (1 degree of freedom)
Blue - Gaussian
We can use gradient methods to find an embedding

Cost function

\[ C = KL(P || Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \]

- Large \( p_{ij} \) modeled by small \( q_{ij} \): Large penalty
- Small \( p_{ij} \) modeled by large \( q_{ij} \): Small penalty
- t-SNE mainly preserves local similarity structure of the data

Gradient

\[ \frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + \|y_i - y_j\|^2)^{-1}(y_i - y_j) \]

\( p_{ij} \) = New (low) dimension distance
\( q_{ij} \) = Original (high) dimension D

(not okay to bring distant points closer)

(okay to separate nearby points)
Interpretation of SNE (left) and t-SNE (right) gradients
t-SNE of MNIST digits
Today: Gene Expression, PCA, t-SNE, autoencoders

- Gene expression analysis: The Biology of RNA-seq
- Supervised (Classification) vs. unsupervised (Clustering)
- Supervised: Differential expression analysis
- Unsupervised: Embedding into lower dimensional space
- Linear reduction of dimensionality
  - Principle Component Analysis
  - Singular Value Decomposition
- Non-linear dimensionality reduction: embeddings
  - t-distributed Stochastic Network Embedding (t-SNE)
  - Building intuition: Playing with t-SNE parameters
- Deep Learning embeddings
  - Autoencoders
7. Playing with t-SNE parameters
Perplexity matters

Recommended range by Van Der Maaten and Hinton

https://distill.pub/2016/misread-tsne/
Number of steps matter

Original

Perplexity: 30
Step: 10

Perplexity: 30
Step: 20

Perplexity: 30
Step: 60

Perplexity: 30
Step: 120

Perplexity: 30
Step: 1,000

“pinched”: Not enough steps

Too tight

Spread again

Tight again

https://distill.pub/2016/misread-tsne/
Cluster sizes are not meaningful

Original data: 2 Gaussians
Widely different (10-fold) dispersion

t-SNE loses that notion of distance.
By design, it adapts to regional variations in distance.

https://distill.pub/2016/misread-tsne/
Between-cluster distance is not always preserved
False clusters may appear
Relationships are not always preserved
Different runs may produce similar results...

(or not at very low perplexity)

https://distill.pub/2016/misread-tsne/
t-SNE of equidistant points
t-SNE of square grid
t-SNE of 3D Knot
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space
• Linear reduction of dimensionality
  — Principle Component Analysis
  — Singular Value Decomposition
• Non-linear dimensionality reduction: embeddings
  — t-distributed Stochastic Network Embedding (t-SNE)
  — Building intuition: Playing with t-SNE parameters
• Deep Learning embeddings
  — Autoencoders
8. Embedding with Deep learning: Auto-encoders
Autoencoder: dimensionality reduction with neural net

- Tricking a **supervised** learning algorithm to work in **unsupervised** fashion
- Feed input as output function to be learned. **But!** Constrain model complexity

- **Pretraining** with RBMs to learn representations for future supervised tasks. Use RBM output as “data” for training the next layer in stack
- After pretraining, "unroll" RBMs to create deep autoencoder
- Fine-tune using backpropagation

[Hinton et al, 2006]
Autoencoders learn a latent representation of data

\[ L(x, g(f(x))) \]
Denoising autoencoders recover signal corrupted by noise

\[ L(x, g(f(\tilde{x}))) \]
We can learn manifolds with autoencoders
Auto-encoder learning of MNIST digit data

2D codes from a 784-300-300-2-300-300-784 autoencoder (RBM pretrained + finetuned):

Decoded digit:

Pos X/Y: 77.28, -79.6000
Mouse X/Y: 789, 408
Offset X/Y: -453, -209
Scale X/Y: 0.23, -0.4

Pos X = (Mouse X + Offset X) * Scale X
Pos Y = (Mouse Y + Offset Y) * Scale Y

http://elf-project.sourceforge.net/autoencoder.html
Today: Gene Expression, PCA, t-SNE, autoencoders

• Gene expression analysis: The Biology of RNA-seq
• Supervised (Classification) vs. unsupervised (Clustering)
• Supervised: Differential expression analysis
• Unsupervised: Embedding into lower dimensional space
• Linear reduction of dimensionality
  — Principle Component Analysis
  — Singular Value Decomposition
• Non-linear dimensionality reduction: embeddings
  — t-distributed Stochastic Network Embedding (t-SNE)
• Deep Learning embeddings
  — Autoencoders
FIN - Thank You
Interesting on-line demos

http://dpkingma.com/sgvb_mnist_demo/demo_old.html
http://elf-project.sourceforge.net/autoencoder.html
http://vdumoulin.github.io/morphing_faces/online_demo.html