CSC 411 Lecture 13: t-SNE

Ethan Fetaya, James Lucas and Emad Andrews

University of Toronto
Today

- SNE - Stochastic Neighbor Embedding
- t-SNE
- KL divergence
Local embedding

- t-SNE is an alternative dimensionality reduction algorithm.
- PCA tries to find a global structure
  - Low dimensional subspace
  - Can lead to local inconsistencies
    - Far away point can become nearest neighbors
- t-SNE tries to preserve local structure
  - Low dimensional neighborhood should be the same as original neighborhood.
- Unlike PCA almost only used for visualization
  - No easy way to embed new points
tSNE 2 dimensions embedding for MNIST
PCA 2 dimensions embedding for MNIST
Stochastic Neighbor Embedding (SNE)

SNE basic idea:

- "Encode" high dimensional neighborhood information as a distribution
- Intuition: Random walk between data points.
  - High probability to jump to a close point
- Find low dimensional points such that their neighborhood distribution is similar.
- How do you measure distance between distributions?
  - Most common measure: KL divergence
Consider the neighborhood around an input data point $x_i \in \mathbb{R}^d$

Imagine that we have a Gaussian distribution centered around $x_i$

Then the probability that $x_i$ chooses some other datapoint $x_j$ as its neighbor is in proportion with the density under this Gaussian

A point closer to $x_i$ will be more likely than one further away
Proporties $P_{ij}$

The $i \rightarrow j$ probability (should be familiar from A1Q2), is the probability that point $x_i$ chooses $x_j$ as its neighbor

$$P_{j|i} = \frac{\exp \left( -||x^{(i)} - x^{(j)}||^2 / 2\sigma_i^2 \right)}{\sum_{k \neq i} \exp \left( -||x^{(i)} - x^{(k)}||^2 / 2\sigma_i^2 \right)}$$

With $P_{i|i} = 0$

- The parameter $\sigma_i$ sets the size of the neighborhood
  - Very low $\sigma_i$ - all the probability is in the nearest neighbor.
  - Very high $\sigma_i$ - Uniform weights.

- Here we set $\sigma_i$ differently for each data point

- Results depend heavily on $\sigma_i$ - it defines the neighborhoods we are trying to preserve.

- Final distribution over pairs is symmetrized: $P_{ij} = \frac{1}{2N} (P_{i|j} + P_{j|i})$
  - Pick $i$ (or $j$) uniformly and then "jump" to $j$ ($i$) according to $P_{j|i}$ ($P_{i|j}$)
For each distribution $P_{j|i}$ (depends on $\sigma_i$) we define the perplexity

\[ \text{perp}(P_{j|i}) = 2^{H(P_{j|i})} \]

where $H(P) = -\sum_i P_i \log(P_i)$ is the entropy.

If $P$ is uniform over $k$ elements - perplexity is $k$.

- Smooth version of $k$ in $kNN$
- Low perplexity = small $\sigma^2$
- High perplexity = large $\sigma^2$

Define the desired perplexity and set $\sigma_i$ to get that (bisection method)

Values between 5-50 usually work well

Important parameter - different perplexity can capture different scales in the data

If your interested - try A1Q2 which a fixed perplexity instead (let me know how it worked)!
Perplexity

[Pic credit: https://distill.pub/2016/misread-tsne/]
SNE objective

- Given \( \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)} \in \mathbb{R}^D \) we define the distribution \( P_{ij} \)
- Goal: Find good embedding \( \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)} \in \mathbb{R}^d \) for some \( d < D \) (normally 2 or 3)
- How do we measure an embedding quality?
- For points \( \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)} \in \mathbb{R}^d \) we can define distribution \( Q \) similarly the same (notice no \( \sigma_i^2 \) and not symmetric)

\[
Q_{ij} = \frac{\exp\left(-||\mathbf{y}^{(i)} - \mathbf{y}^{(j)}||^2\right)}{\sum_k \sum_{l \neq k} \exp\left(-||\mathbf{y}^{(l)} - \mathbf{y}^{(k)}||^2\right)}
\]

- Optimize \( Q \) to be close to \( P \)
  - Minimize KL-divergence
- The embeddings \( \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)} \in \mathbb{R}^d \) are the parameters we are optimizing.
  - How do you embed a new point? No embedding function!
KL divergence

Measures distance between two distributions, $P$ and $Q$:

$$KL(Q\|P) = \sum_{ij} Q_{ij} \log \left( \frac{Q_{ij}}{P_{ij}} \right)$$

- Not a metric function - not symmetric!
- Code theory intuition: If we are transmitting information that is distributed according to $P$, then the optimal (lossless) compression will need to send on average $H(P)$ bits.
- What happens you expect $P$ (and design your compression accordingly) but the actual distribution is $Q$?
  - will send on average $H(Q) + KL(Q\|P)$
  - $KL(Q\|P)$ is the "penalty" for using wrong distribution
KL Properties

- $KL(Q\|P) \geq 0$ and zero only when $Q = P$ (a.s)
- $KL(Q\|P)$ is a convex function.
- if $P_{ij} = 0$ but $Q_{ij} > 0$ then $KL(Q\|P) = \infty$

[Pic credit: https://timvieira.github.io/blog/post/2014/10/06/kl-divergence-as-an-objective-function/]
We have $P$, and are looking for $y^{(1)}, \ldots, y^{(N)} \in \mathbb{R}^d$ such that the distribution $Q$ we infer will minimize $L(Q) = KL(P \| Q)$ (notice $Q$ on right, uncommon).

Note that $KL(P \| Q) = \sum_{ij} P_{ij} \log \left( \frac{P_{ij}}{Q_{ij}} \right) = -\sum_{ij} P_{ij} \log \left( Q_{ij} \right) + \text{const}$

Can show that $\frac{\partial L}{\partial y^{(i)}} = \sum_{j} (P_{ij} - Q_{ij})(y^{(i)} - y^{(j)})$

Not a convex problem! No guarantees, can use multiple restarts.

Main issue - crowding problem.
Crowding Problem

- In high dimension we have more room, points can have a lot of different neighbors.

- In 2D a point can have a few neighbors at distance one all far from each other - what happens when we embed in 1D?

- This is the "crowding problem" - we don’t have enough room to accommodate all neighbors.

- This is one of the biggest problems with SNE.

- t-SNE solution: Change the Gaussian in $Q$ to a heavy tailed distribution.
  - if $Q$ changes slower, we have more "wiggle room" to place points at.
t-SNE

t-Distributed Stochastic Neighbor Embedding

- Student-t Probability density $p(x) \propto (1 + \frac{x^2}{\nu})^{-(\nu+1)/2}$
  - for $\nu = 1$ we get $p(x) \propto \frac{1}{1+x^2}$
- Probability goes to zero much slower then a Gaussian.
- Can show it is equivalent to averaging Gaussians with some prior over $\sigma^2$
- We can now redefine $Q_{ij}$ as

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

- We leave $P_{ij}$ as is!
t-SNE gradients

- Can show that the gradients of t-SNE objective are

\[
\frac{\partial L}{\partial y(i)} = \sum_j (P_{ij} - Q_{ij})(y(i) - y(j))(1 + ||y_i - y_j||^2)^{-1}
\]

- Compare to the SNE gradients: \( \frac{\partial L}{\partial y(i)} = \sum_j (P_{ij} - Q_{ij})(y(i) - y(j)) \)

- Both repulse close dissimilar points and attract far similar points, but the \( t - SNE \) has a smaller attraction term to solve crowding.

[Image credit: "Visualizing Data using t-SNE"]
Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

**Data:** data set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations $T$, learning rate $\eta$, momentum $\alpha(t)$.

**Result:** low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.

**begin**

compute pairwise affinities $p_{ji}$ with perplexity $Perp$ (using Equation 1)

set $p_{ij} = \frac{p_{ji} + p_{ij}}{2n}$

sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$

for $t=1$ to $T$ do

compute low-dimensional affinities $q_{ij}$ (using Equation 4)

compute gradient $\frac{\delta C}{\delta y}$ (using Equation 5)

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

end

**end**

[Slide credit: "Visualizing Data using t-SNE"]
CNN features example

[Image credit: http://cs.stanford.edu/people/karpathy/cnnembed/]
CNN features example

[Image credit: https://lvdmaaten.github.io/tsne/]
Recap

- t-SNE is a great way to visualize data
- Helps understand "black-box" algorithms like DNN.
- Reduced "crowding problem" with heavy tailed distribution.
- Non-convex optimization - solved by GD with momentum.
- Less suitable for SGD (think about the parameters), some alternative speedups exist ("Barnes-Hut t-SNE").
- Great extra resource: https://distill.pub/2016/misread-tsne/