Tensor Distribution
Sensitive Hashing

Mihir Mongia, Benjamin Soudry, Arash Gholami Davoodi, Professor Hosein Mohimani

Computational Biology Department
School of Computer Science
Carnegie Mellon University
Proteins, Proteins, Proteins!!!

- Cell:Earth::Proteins:Machines
- Messenger Proteins: Growth Hormone
- Enzymes
- Transport Proteins:Ferritin
A key difference though: We can’t engineer them

- We can’t make protein to prevent wrinkles
- Or proteins that eat up fat
- Or more importantly perhaps proteins that could eat nuclear waste

- Because we don’t understand them
Can barely even predict their structure

• Proteins are at the most basic level a string with 20 character alphabet.

• Given a sequence we have extremely poor tools to predict structure...so poor people now resort to deep learning

AlphaFold: Using AI for scientific discovery
So we need to study proteins further

• If you were to ever understand what proteins do you may want to understand what proteins exist in a particular ecosystem and then associate it to the properties of the ecosystem

• Hence Mass Spectrometry
A mathematical formulation of the biological problem

Let \( G \) be a generative process that generates pairs of vectors \((x_i, y_i)\) from a discrete joint distribution. Each position in the pair \((x_i, y_i)\) is generated independently of any other position.

**Example:** Let the elements of \( x_i \) come from the alphabet = \{a,b\}. Let the elements of \( y_i \) come from the alphabet = \{0,1\}. The pairs come from the following discrete distribution:

\[
\begin{pmatrix}
  a \\
  b \\
\end{pmatrix}
= \begin{bmatrix}
  .4 & .1 \\
  .1 & .4 \\
\end{bmatrix}
\]

Then a possible pair of \((x_i, y_i)\) that might be jointly generated from \( G \) might be the following

\[
y_i = \text{a a b a b a a b b}
x_i = 0 0 1 0 1 1 0 0 1
\]
A mathematical formulation of the biological problem

Suppose then we generate two data sets $X$ and $Y$ where each row of $X$ and $Y$ is generated from $G$ in the previous slide. After generating the two data sets, let an oracle generate $X'$ and $Y'$ by scrambling the rows of $X$ and $Y$ in some unknown fashion, so the rows of $X'$ and $Y'$ no longer correspond to each other.
A mathematical formulation of the biological problem

But the scrambling order is hidden from us

Knowing the entries of $X'$ and $Y'$ and the generative process $G$, is there a fast algorithm to figure out for each row of $X'$ what row $Y'$ corresponds to it?
The naïve approach – law of large numbers

First note that in the unscrambled data set the for i not equal to j, \(x_i\) is uncorrelated \(y_j\). Thus the joint distribution between \(x_i\) and \(y_j\) can be calculated by looking at the marginals of.

**Example:** Take the same distribution we had before.

\[ P = \begin{bmatrix} 0 & 1 \\ a & b \end{bmatrix} \begin{bmatrix} .4 & .1 \\ .1 & .4 \end{bmatrix} \]

\(x_i\) and \(y_j\) come from the distribution

\[ Q = \begin{bmatrix} 0 & 1 \\ a & b \end{bmatrix} \begin{bmatrix} .5 & .5 \\ .5 & .5 \end{bmatrix} \]
The naïve approach – law of large numbers

Thus each pair $x_i$ and $y_j$ will satisfy the law of large numbers of for either of the two distributions.

Example: Take the same distribution we had before.

$$P = \begin{bmatrix} .4 & .1 \\ .1 & .4 \end{bmatrix} \quad Q = \begin{bmatrix} .5 & .5 \\ .5 & .5 \end{bmatrix}$$

$y_1 = a \ b \ a \ b \ b \ a \ a \ b \ b$

$x_1 = 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1$

$y_2 = a \ a \ b \ a \ b \ b \ a \ a \ b \ b$

$x_2 = 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1$

Thus we can look at each pair $i$ and $j$ and see if the law of large numbers is satisfied for either $P$ or $Q$. But this takes $O(n^2)$ work, where $n$ is the length of $X$ and $Y$. 
A detour

Uniform hashing families

The computing problem

The solution
Taking inspiration from nearest neighbor search

Again we have datasets X and Y but this time they have the same alphabet and the goal is to find for each $x_i$ the $y_j$ that is closest to it.

**Example:** Hamming distance between binary vectors. Hamming distance between two vectors is number of positions where two binary vectors disagree.

\[
X = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}
\]
\[
Y = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}
\]

How find nearest neighbors in a faster than $O(n^2)$?
Send data points to buckets in a strategic way

In the naïve approach we simply compare all pair of points. Thus we can imagine the amount of computation needed to be a square.

But what if we send \(y_1, y_2, x_1, x_2\) to bucket 1, and \(y_3, y_4, x_3, x_4\) to bucket 2. Then if we just compare the elements inside the individual buckets?

16 comparisons

8 comparisons
We just need one key idea – universal hashing

Let $H = \{h_1, h_2, h_3, ..., h_n\}$ be a set of functions with the following property: If you pick a function $h$ from $H$ uniformly randomly then

- If $x_i$ and $y_j$ satisfy some condition Desired, then $P(h(x_i) = h(y_j)) > \alpha$
- If $x_i$ and $y_j$ satisfy some condition Undesired, then $P(h(x_i) = h(y_j)) < \beta$
What is the complexity?

\[ N^{\frac{\log(\alpha)}{\log(\beta)}} \]
So what if our desired conditions dealt with a distribution

- If $x_i$ and $y_j$ are generated from $P$, then $P(h_x(x_i) = h_y(y_j)) > \alpha$
- If $x_i$ and $y_j$ are generated from $Q$, then $P(h_x(x_i) = h_y(y_j)) < \beta$

How do we create such a thing?
Go back to example

$$P = \begin{pmatrix} a \\ b \end{pmatrix} \begin{bmatrix} .4 & .1 \\ .1 & .4 \end{bmatrix}$$

Pick position at random.

$$H_{a,b}$$

$$H_{0,1}$$
Hashes are equivalent to matrices!
• If $x_i$ and $y_j$ are generated from $P$, then $P(h_x(x_i) = h_y(y_j)) > \alpha$
• If $x_i$ and $y_j$ are generated from $Q$, then $P(h_x(x_i) = h_y(y_j)) < \beta$

\[
H_{a,b} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

\[
H_{0,1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

\[
\text{Trace}(H_{a,b}^T PH_{0,1}) = \alpha
\]

\[
\text{Trace}(H_{a,b}^T QH_{0,1}) = \beta
\]
But would could pick more than one position, we could pick two or three or more

\[
\text{Trace}(H_{a,b}^T P \otimes P \otimes P \ldots H_{0,1}) = \alpha
\]

\[
\text{Trace}(H_{a,b}^T Q \otimes Q \otimes Q \ldots H_{0,1}) = \beta
\]
Just need to optimize

\[ \text{Trace}(H_{a,b}^T P \otimes P \otimes P \ldots H_{0,1}) = \alpha \]

\[ \text{Trace}(H_{a,b}^T Q \otimes Q \otimes Q \ldots H_{0,1}) = \beta \]

**Integer Linear Program to make alpha high and beta low.** Remember complexity is roughly

\[ N \frac{\log(\alpha)}{\log(\beta)} \]
Results