Homework 1 example solution

Compiled by Nikita Alexandrov

October 19 2018

Question 1

The proposed distance function does not hold the following metric properties:

- **Isolation**: if we choose any non-zero vector $\mathbf{x}$ and any $a > 0$, then $\cos(\mathbf{x}, a\mathbf{x}) = \frac{(\mathbf{x}, a\mathbf{x})}{||\mathbf{x}||||a\mathbf{x}||} = \frac{a(\mathbf{x}, \mathbf{x})}{a||\mathbf{x}||||\mathbf{x}||} = \frac{||\mathbf{x}||^2}{||\mathbf{x}||^2} = = 1 \Rightarrow d_{cos}(\mathbf{x}, a\mathbf{x}) = 0$, however, $\mathbf{x} \neq a\mathbf{x}$
- **Triangle inequality**: let's consider $\mathbf{x} = (1, 0, 0, \ldots, 0)^T$, $\mathbf{y} = (0, 1, 0, \ldots, 0)^T$, $\mathbf{z} = (\sqrt{2}, \sqrt{2}, 0, \ldots, 0)^T$. As can be verified, $\cos(\mathbf{x}, \mathbf{y}) = 0, \cos(\mathbf{x}, \mathbf{z}) = \cos(\mathbf{y}, \mathbf{z}) = \frac{\sqrt{2}}{2}$. Consequently,

  $d_{cos}(\mathbf{x}, \mathbf{z}) + d_{cos}(\mathbf{z}, \mathbf{y}) = 1 - \frac{\cos(\mathbf{x}, \mathbf{z}) + \cos(\mathbf{y}, \mathbf{z})}{2} = 1 - \frac{\sqrt{2}}{2} < 1 = d_{cos}(\mathbf{x}, \mathbf{y})$.

Any counterexample that disproves any property is sufficient to show that d_{cos} isn't a metric.

Question 2

2.1 Is D a metric?

Indeed, $D(x, y)$ is a metric and let's prove this fact for any $k > 0$. We know that d is a metric, therefore d satisfies all mentioned below properties.

Non-negativity If $D(x, y) \leq 0$, because the $d(x, y) > 0$.

Isolation If $x = y$, then $d(x, y) = 0 \Rightarrow D(x, y) = 0$ and at the same time, if $D(x, y) = 0$, then $d(x, y) = 0 \Rightarrow x = y$.

Symmetry Symmetry holds from the fact that $d(x, y)$ is symmetrical:
$d(x, y) = d(y, x) \Rightarrow D(x, y) = \frac{d(x, y)}{d(x, y) + k} = \frac{d(y, x)}{d(y, x) + k} = D(y, x)$

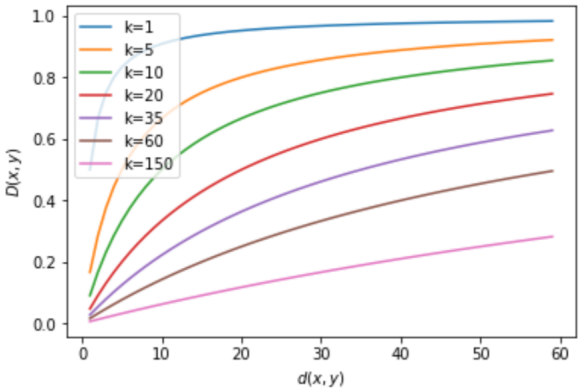
Triangle inequality If we denote $f(p) = \frac{p}{p + k}$, it is very easy to show that if $p \leq q$, then $f(p) \leq f(q)$:
$p \leq q \Rightarrow (q + k)p = pk + pq \leq qk + pq = (p + k)q \Rightarrow \frac{p}{p + k} \leq \frac{q}{q + k}$.
(Or we could just say that the derivative of f is positive for any p: $f'(p) = \frac{k}{(p + k)^2}$).
Since, $d(x, y) \leq d(x, z) + d(z, y)$, then $D(x, y) = f(d(x, y)) \leq f(d(x, z) + d(z, y))$. At the same time,

$$f(d(x, z) + d(z, y)) = \frac{d(x, z) + d(z, y)}{d(x, z) + d(z, y) + k} = \frac{d(x, z)}{d(x, z) + d(z, y) + k} + \frac{d(z, y)}{d(x, z) + d(z, y) + k} \leq$$
$$\leq \frac{d(x, z)}{d(x, z) + k} + \frac{d(z, y)}{d(z, y) + k} = D(x, z) + D(z, y)$$

Therefore, $D(x, y) \leq D(x, z) + D(z, y)$ and we proved that $D(x, y)$ is a metric.

2.2 What is the role of k?

k parameter affects the shape of the new metric curve as follows (credits for picture: Dimitrios Papatheodorou):



Now we can see that if k is close to 0, the new metric grows very fast as a function of $d(x, y)$ and close to 1 even for small values of $d(x, y)$. On the other hand, if k is very large, this function grows very slow.

2.3 What is the possible application of the new metric?

The possible values of the new metric always lie on the interval $[0, 1)$. It can be useful to define the similarity function between x and y as $s(x, y) = 1 - D(x, y)$. Also it calibrates the previous distance function $d(x, y)$ in accordance with the value k: if k is close to 0 and $D(x, y)$ is small, it means that $d(x, y)$ was very small.

Question 3

Non-negativity Since (X, d) is a metric space, $d_H(A, B) = \max_{x \in A} \min_{y \in B} d(x, y)$ is non-negative.

Symmetry Symmetry holds because of the definition of $d'_H(A, B) = \max\{d_H(A, B), d_H(B, A)\}$

Isolation $A = B \to d'_H(A, B) = 0$ is obvious. $d'_H(A, B) = 0$ implies that $A \subseteq B$ and $B \subseteq A$. Thus, $A = B$.

Triangle inequality For $\forall a \in A$

$$
\begin{aligned}
d_H(a, B) &= \min_{b \in B} d(a, b) \\
&\leq \min_{b \in B}(d(a, c) + d(c, b)) && \forall c \in C \\
&= d_H(a, c) + \min_{b \in B} d(c, b) && \forall c \in C \\
&\leq d(a, c) + d_H(c, B) && \forall c \in C \\
&\leq d(a, c) + d_H(C, B) && \forall c \in C
\end{aligned}
$$

$$
\begin{aligned}
d_H(a, B) = \min_{c \in C} d_H(a, B) &= \min_{c \in C}(d_H(a, c) + d_H(C, B)) \\
&\leq d_H(a, C) + d_H(C, B)
\end{aligned}
$$

$$
\begin{aligned}
d_H(A, B) = \max_{a \in A} d_H(a, B) &\leq \max_{a \in A}(d_H(a, C) + d_H(C, B)) \\
&\leq d_H(A, C) + d_H(C, B))
\end{aligned}
$$

$d_H(B, A) \leq d_H(B, C) + d_H(C, A)$ can be proved in a similar way.
Thus, $d'_H(A, B) \leq d'_H(A, C) + d'_H(B, C)$
Therefore, d'_H is a metric.

Question 4

4.1 Propose a distance function for two walks

Let's consider the sequence of vertices $A = \{a_0, a_1, \ldots, a_k\}$ and $B = \{b_0, b_1, \ldots, b_l\}$ as a string $a = a_0 a_1 \ldots a_k$ and $b = b_0 b_1 \ldots b_l$. Then the distance function between A and B can be defined as the edit distance between a and b.

Of course, there are many other correct ways to define a distance function, but we will use the proposed function in the next questions.

4.2 Show the intuition of the distance function

- The distance function must preserve the order between the vertices or else $A = \{a, b, c, d\}$ would be the same as a walk $B = \{a, c, b, d\}$ and the edit distance treats these strings as non-equivalent.
- Edit distance can work with walks of different lengths and there are no restrictions on the walks.
- The edit distance also works great, when two walks overlap a lot: in case when $A = \{a, b, c, d, \ldots, e\}$ and $B = \{b, c, d, \ldots, e, f\}$, the distance between them will be small: only two operations are needed to remove a and add f to the end of string.

4.3 Show (or disprove) that the proposed distance function is a metric

Let's prove that the edit distance function is a metric.

Non-negativity Edit distance is measured in the number of operations required, it cannot be negative.

Isolation If $A = B$, then there is no need to perform any operations to transform A into B, so $d(A, B) = 0$. On the other hand if implies that $A \neq B$ we need at least one operation to make them equal, therefore $d(A, B) = 0 \Rightarrow A = B$.

Symmetry Operations required to transform the string A into B can be reverted and the reversed sequence transforms B into A. Consequently, $d(A, B) = d(B, A)$.

Triangle Inequality If we have strings A, B, C and if we assume that $d(A, C) > d(A, B) + d(B, C)$, at the beginning we can transform A into B and after that, B into C. It will take $d(A, B) + d(B, C)$ operations, but this sequence transformed A into C. Contradiction. Therefore, $d(A, C) \leq d(A, B) + d(B, C)$.

4.4 Provide an algorithm to compute the distance function

Let $d_{i,j}$ be the edit distance between substrings $a_1 a_2 ... a_i$ and $b_1 b_2 ... b_j$. Then, d_{ij} can be defined by the recurrence:

$$d_{i,j} = \begin{cases} d_{i-1,j-1} & \text{if } a_i = b_j \\ min(d_{i,j-1}, d_{i-1,j}, d_{i-1,j-1}) + 1 & \text{if } a_i \neq b_j. \end{cases}$$

The logic behind the formula is clear:

- When the last symbols are the same, we can remove them and solve the problem for strings $a_1 a_2 ... a_{i-1}$ and $b_1 b_2 ... b_{j-1}$ and the answer is $d_{i-1,j-1}$
- In case when the last symbols differ:
  - We can use $d_{i,j-1}$ to transform $a_1 a_2 ... a_i$ into $b_1 b_2 ... b_{j-1}$ and b_j to the end.
  - We can remove a_i and transform $a_1 a_2 ... a_{i-1}$ into $b_1 b_2 ... b_j$ with $d_{i-1,j}$ operations.
  - We can substitute a_i by b_j and transform $a_1 a_2 ... a_{i-1}$ into $b_1 b_2 ... b_{j-1}$ with $d_{i-1,j-1}$ operations.

Also we need to define the cases when $i = 0$ or $j = 0$: $d_{0,j} = j$ and $d_{i,0} = i$.

Our answer will be $d_{|A|,|B|}$ where $|A|$ and $|B|$ are the length of the walks (or corresponding strings). This algorithm requires $\mathcal{O}(|A||B|)$ time, because we calculate $d_{i,j}, \forall i = 0, \ldots, |A|$ and $j = 0, \ldots, |B|$

4.5 Propose an appropriate distance function on the 2-d space

We can represent the walk X through a 2-d space as 2 functions of time such as $x_1(t)$ and $x_2(t)$, where $0 \leq t \leq ||X||$, where $||X||$ is a walk length. Basically the walk is the composition of two time series functions.

Dynamic Time Warping distance which was discussed on the lectures, can be one of the options. DTW algorithm warps the two functions to find a match from one point in a function to multiple points to the other function. Please note that DTW distance isn't the metric.

Credits: Ananth Mahadevan

Question 5

L_∞ is the maximum of the difference between two elements in one of the d dimensions. We calculate the maximum and the minimum value of each dimension, and then subtract them, which takes $O(dn)$. After that, we loop through d dimensions to get the maximum value among these subtractions, which corresponds to the furthest pair.

The basic idea is shown as follows:

$$\begin{aligned}
\max_{i,j \in \{1,\ldots,n\}} L_\infty(x_i, x_j) &= \max_{i,j \in \{1,\ldots,n\}} \max_{t \in \{1,\ldots,d\}} |x_{i,t} - x_{j,t}| \\
&= \max_{t \in \{1,\ldots,d\}} \left\{ \max_{i,j \in \{1,\ldots,n\}} |x_{i,t} - x_{j,t}| \right\} \\
&= \max_{t \in \{1,\ldots,d\}} \left\{ \max_{i \in \{1,\ldots,n\}} x_{i,t} - \min_{j \in \{1,\ldots,n\}} x_{j,t} \right\}
\end{aligned}$$

Question 6

6.1 How can d_l be used to speed up the task of finding the nearest object?

If we process the object x_i, it can be noticed that if the value of lower-bound distance $d_l(x_i, q)$ is no less than the current minimum $dmin$, there is no sense to calculate $d(x_i, q)$ since $d(x_i, q) \geq d_l(x_i, q) \geq dmin$ and eventually, the minimum will not be updated. Therefore, by comparing $d_l(x_i, q)$ with the current minimum, we can take out of consideration the element and we will save time in this case, because d_l is much faster to compute than d.

6.2 Pseudocode

Credits: Jaakko Kuusela

```
1: procedure NEAREST WITH LOWER BOUND
2:     dmin ← d(x_1, q)
3:     x* ← x_1
4:     for i = 2,...,n do
5:         dtemp ← d_l(x_i, q)
6:         if dtemp < dmin then
7:             dtemp ← d(x_i, q)
8:             if dtemp < dmin then
9:                 dmin ← dtemp
10:                x* ← x_i
11:            end if
12:        end if
13:    end for
14:    return x*
15: end procedure
```

6.3 Why is it important to have the lower-bound distance function as large as possible?

We don't calculate "expensive" distance function d in case when $d_l(x_i, q) \geq dmin$. Therefore when we have larger values of d_l, we are increasing the chance that the current object is definitely not a minimum. For example, the lower-bound distance function $d_l(x, y) = 0$ isn't helpful at all: we will have to calculate $d(x_i, q)\, \forall i$.

6.4 Lower-bound distance function for string edit distance

A lower bound can be given as $d_l(x, y) = ||x| - |y||$, where $|x|$ and $|y|$ are the length of strings x and y.

$d_l(x, y) \leq d(x, y)$ because we need to perform at least $|y| - |x|$ "insert" operations (if x is shorter than y) or to complete at least $|x| - |y|$ "delete" operations (if x is longer than y).