Homework 2: Similarity measures on time series and graphs

Theresa Wakonig
Data Mining I

October 28, 2020

1 Exercise 1

Analyze 'ECG' heartbeat time series.

1.a

See *compute_dtw.py*.

Generated results for average DTW and Manhattan distances between groups:

Pair of classes	Manhattan	DTW, w = 0	DTW, w = 10	DTW, w = 25	DTW, w = inf
abnormal:abnormal	67.77	67.77	38.65	26.48	25.37
abnormal:normal	67.52	67.52	34.20	26.94	26.35
normal:normal	45.65	45.65	24.42	22.17	21.87

1.b

For w = 0, we are just moving along the diagonal and are thus only summing up the base elementwise distances (Manhattan). This is equal to treating the time series as vectors and calculating the Manhattan distance. The results above correctly depict the behaviour of this special case.
Intuitively, distances between measurements of the same groups should be smaller than when compared to measurements of a different group. The abnormal:abnormal comparison yields a rather big value for small w, which might be due to the fact that a heartbeat can be "abnormal" in many different ways and might not be as clearly defined as the "normal" heartbeat classification.
I would choose the DTW distance with a relatively big window. The case w $\geq$ 25 seems favourable for this example, as comparisons among the same groups yield smaller distances than mixed comparisons.

1.c

The bigger the hyperparameter w (window size), the smaller the computed distances. The value of the parameter w defines a max. horizontal distance from the diagonal of the DTW matrix.

Larger w $\rightarrow$ more comparisons between values of time series $\rightarrow$ runtime increases

1.d

No, the dynamic time warping distance is not a metric as it does not fulfill the triangle inequality and property 2 (metric is 0 iff $x_1 = x_2$).

E.g.: time series x, y, z; base distance d(i, j); $\rightarrow$ Manhattan; DTW definition same as slide 50

$x = [0, 2, 2, 3]$
$y = [0, 2, 3]$
$z = [0, 3, 3]$

$DTW(x, y) = 0, x \neq y$

$DTW(x, z) = 2 \nleq 1 = DTW(x, y) + DTW(y, z)$

1.e

Two for loops iterating over matrix (n x n) elements.

w-constrained warping: $O(nw)$
w-constrained warping: $O(n^2)$

2 Exercise 2

Analyze 'MUTAG' dataset.

2.a

See *shortest_path_kernel.py*.

2.b

See *compute_sp_kernel.py*.

Very similar results but different runtime (use symmetric property, because we are only referring to undirected graphs).

Pair of classes	SP
mutagenic:mutagenic	5309.92
mutagenic:non-mutagenic	2706.78
non-mutagenic:non-mutagenic	1433.28

Table 1: Average similarities for comparing all elements of a group with each other (include sp_kernel(S_i, S_j) and sp_kernel(S_j, S_i)).

Pair of classes	SP
mutagenic:mutagenic	5316.09
mutagenic:non-mutagenic	2642.57
non-mutagenic:non-mutagenic	1438.15

Table 2: Average similarities using symmetry: sp_kernel(S_i, S_j) = sp_kernel(S_j, S_i). This leads to less iterations and a smaller runtime.

2.c

Floyd-Warshall: Three for-loops, each iterating through all nodes $\rightarrow O(n^3)$

SP-Kernel: For directed graphs, a number of n^2 edges have to be considered per graph. Comparing two graphs therefore yields a complexity of $O(n^4)$. In our case we only look at undirected graphs. Therefore, our SP matrices will always be symmetric and the edge walks as well. As proposed in 2b and *compute_sp_kernel.py* one must not run through/compare all the entries but just $\frac{n^2-n}{2}$ (triangular part without diagonal).