1 Definition of statistical indistinguishability(SI)

For simplicity, assume alphabet size for all columns are equal to k and the number of columns is d. We say that the co-occurrence matrix C of dimension d is statistically indistinguishable if for any marginal co-occurrence matrix and C, the following holds: there exists a (disjoint) partition $\{E_i\}$ on the marginal attributes such that:

1. $|E_i| = k$,
2. If $E_i = \{e_0, e_1, \cdots, e_{k-1}\}$, then columns of e_j is a permutation of $\{0, 1, \cdots, k-1\}$. Furthermore, the counts on E_i are all equal.

2 Simple case

We know very well how to pad a database with two columns so that alphabets in each attribute column is indistinguishable: for every attribute in the first row of co-occurrence matrix (k by k), find a feasible permutation (this need to be defined carefully in a full formal setting), and assign all elements to the same count.

To verify the SI property, we need to check the co-occurrence matrix and the marginal co-occurrence matrices:

1. The co-occurrence matrix itself satisfies the conditions by construction.
2. The marginals in this case are marginal counts over single attribute column. The marginal counts are all equal because all of them contains exactly one attribute from each of the partitions for the co-occurrence matrix.

For example, in the 4 by 4 case, if we denote the entries of the co-occurrence matrix by their partition index, we can have something like this:

$$\begin{array}{cccc} 0 & 2 & 1 & 3 \\ 2 & 3 & 0 & 1 \\ 3 & 1 & 2 & 0 \\ 1 & 0 & 3 & 2 \end{array}$$

The marginal counts will be ‘a sum of 0, 1, 2 and 3’, so all of them are equal.

3 The challenge

However, to achieve SI over three columns of the database is not so easy. This is because if we fix the behaviour over any two attribute column pairs, the third attribute column pair does not necessarily preserve the ‘permutation’ property.

For example, if we have 3 columns each has alphabet size 4. The marginal co-occurrence counts (using the index representation) between first two columns is the one before. The marginal co-occurrence counts between first column and third column is:

$$\begin{array}{cccc} 0 & 1 & 2 & 3 \\ 2 & 3 & 0 & 1 \\ 1 & 2 & 3 & 0 \\ 3 & 0 & 1 & 2 \end{array}$$

Then the marginal counts over second and third column does not satisfy the requirement.

To see this, we first need to find a representation of marginal counts over second and third column. We can do this by reading off the other two marginal co-occurrence matrices. For example, the marginal count of $(1, 2)$ (over second and third column) is equal to sum of counts in the original co-occurrence matrix, or $\sum_x (x, 1, 2)$. We know how $(x, 1)$ looks like, over first and second column, and how $(x, 2)$ looks like, over first and third column. By putting the constraints together, we get how $(1, 2)$ looks like.

We write constraints on second and third column as a sum over $[x, y]$ where x and y are entries in the matrices above. This is a valid way to construct marginal co-occurrence matrix over second and third column because actual attributes need to satisfy both constraints.

Then we can write

$$\begin{aligned} (1,2) &= (0,1,2) + (1,1,2) + (2,1,2) + (3,1,2) \\ &= [2,1] + [3,2] + [0,3] + [1,0] \end{aligned}$$

This is simply first row of the two matrices in joint.

This has to equal to $k - 1$ other marginal attribute counts, but this does not happen here. For instance, we need to have $(0, x)$ for some x to have the same count. But

$$\begin{aligned} (0,0) &= [0,0] + [2,1] + [1,2] + [3,3] \\ (0,1) &= [0,2] + [2,3] + [1,0] + [3,1] \\ (0,2) &= [0,1] + [2,2] + [1,3] + [3,0] \\ (0,3) &= [0,3] + [2,0] + [1,1] + [3,2] \end{aligned}$$

A simple counting argument can be used to explain this disparity. Since there are k different rows in each marginal co-occurrence matrix, with two of them, the number of distinct constraints produced by the pair is potentially as large as k^2, but the requirement on the marginal co-occurrence matrix on second and third column basically says there can only be k distinct values, hence the disparity.

The problem then becomes: how to constrain the first two marginal co-occurrence matrices in construction, so that the third marginal co-occurrence matrix has the desired properties.

An unimplemented solution: Suppose instead of having two different marginal co-occurrence matrices between first two columns and first and third column, there is a single marginal co-occurrence matrix, say the first one in the document. Then the desired property indeed holds. Check:

$$(0,0) = (1,1) = (2,2) = (3,3) = [0,0] + [1,1] + [2,2] + [3,3]$$

and so on. Hence it seems that fixing the permutation works. Also, note that we can freely permute the rows in each marginal co-occurrence matrix and get the desired property over the third marginal co-occurrence matrix.

In case the alphabet sizes are not the same, we can make them multiples of each other (so they take the form b^i for a 'base' b) and achieve a similar result by putting additional constraints on the marginal co-occurrence matrices.