Structural Comparison

Eugene Gladyshev

Abstract

This article describes a generic algorithm for comparing patterns.

1 Problem

Given a model vector for example: $\boldsymbol{a} = \{5, 7, 10, 2\}$, assign a similarity score to other vectors of the same size such as, $\boldsymbol{b} = \{0, 5, 7, 3\}$. One way of doing it is to use statistical methods.

- Take the difference vector $\boldsymbol{x} = \sum_{i=0}^{N-1} (b_i - a_i)$.
- Calculate expectation: $\overline{x} = \frac{\sum_{i=0}^{N-1} x_i}{N}$.
- Score as the standard deviation: $score = \sqrt{\frac{\sum_{i=0}^{N-1} (x_i - \overline{x})^2}{N}}$. Where the smallest score indicates the best match.

Obviously, it doesn't work too well for discovering patterns especially for small vectors. If we take a look at the graphs of the model vector $\boldsymbol{a}$ and vector $\boldsymbol{b}$ and another vector $\boldsymbol{c} = \{8, 7, 7, 2\}$, it appears that $\boldsymbol{a}$ and $\boldsymbol{b}$ should produce a better pattern match than $\boldsymbol{a}$ and $\boldsymbol{c}$.

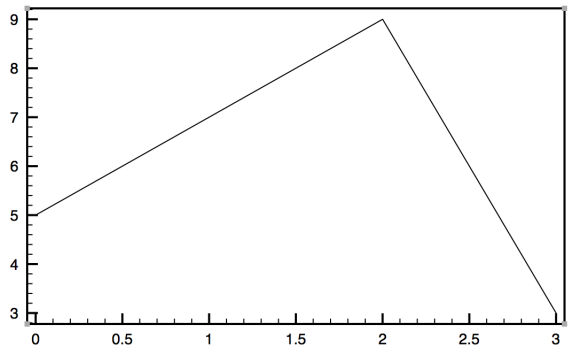

Figure 1: Vector $\boldsymbol{a}$

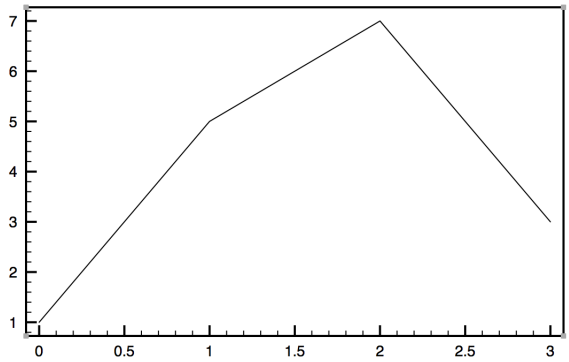

Figure 2: Vector $\boldsymbol{b}$

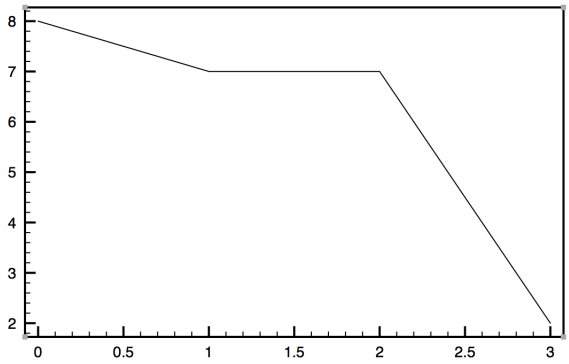

Figure 3: Vector $\boldsymbol{c}$

The statistical method gives the following scores:

- The score for the pair, $\boldsymbol{a}, \boldsymbol{b}$ equal 4.75.
- The score for the pair, $\boldsymbol{a}, \boldsymbol{c}$ equal 3.1875.

It says that $\boldsymbol{a}, \boldsymbol{c}$ is a better match but it doesn't seem very natural.

2 Solution

The simple idea is to score the vector pairs based on their structure instead of differences in values. The process of sorting is employed to reveal the structural similarities.

- Transform the model vector, $\boldsymbol{a}$ into a vector of pairs $\boldsymbol{A} = \{a_i, i\}$. It's a map of elements to their own index.
- Sort $\boldsymbol{A}$ by the first field, a_i.
- Do the same steps with the second vector, $\boldsymbol{b}$. We now have two new vectors, $\boldsymbol{A}$ and $\boldsymbol{B}$.

- For each A_i, take the second field $v_i = secondfield(A_i)$.
- Find v_i in among the second fields in $\boldsymbol{B}$, take the index of the found element in $\boldsymbol{B}$, $x_i = indexof\{B : secondfield(B) = v_i\}$
- We consider i and x_i as coordinates in an N- dimensional space, where $N = sizeof(\boldsymbol{A}) = sizeof(\boldsymbol{B})$.
- The comparison score is the distance between vectors $\boldsymbol{I}$ and $\boldsymbol{X}$, $score = \sqrt{\sum_{i=0}^{N-1}(i - x_i)^2}$.

  The smaller score indicates a better match.

The given method produces the following scores for the three sample vectors: $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$.

- The score for the pair, $\boldsymbol{a}, \boldsymbol{b}$ equal 1.41421.
- The score for the pair, $\boldsymbol{a}, \boldsymbol{c}$ equal 2.44949.

The scores seem more natural indicating $\boldsymbol{a}$ and $\boldsymbol{b}$ as a better match. The structural comparison may be complementary to other methods.

Listing 1: Pseudocode for structural comparison

```
input vectors a, b;

declare vectors A, B

N = sizeof(a);
assert(N = sizeof(b));

for (i=0; i < N; i++) {
  A.add_element({a[i], i});
  B.add_element({b[i], i});
}

sort(A);
sort(B);

score = 0;
for (i = 0; i < N; i++) {
  v = A[i].second_field;
  x = indexof( find_by_second_field(B, v) );
  score = score + (i - x)^2;
}

return score;
```