# Chapter 2
# Fundamentals of Similarity Search

We will now look at the fundamentals of similarity search systems, providing the background for a detailed discussion on similarity search operators in the subsequent chapters. We first look at object representations used in similarity search systems, and then consider attribute types as well as similarity measures used for various attribute types.

## 2.1 Object Representation

As outlined in Chapter 1, similarity search systems most commonly use an attribute-based representation for the dataset of objects to perform similarity search on. Accordingly, a schema would consist of a listing of chosen attributes such as *age*, *name* and *address* along with the domain of values for each of the attributes. A dataset that is based on this schema would have many objects, each object taking a value for each of the specified attributes in the schema; *null* values may or may not be allowed depending on whether the similarity engine can estimate the similarity between a value and a *null*. Thus, a full specification of the data is achieved when the value for each attribute is specified for each object in the dataset. Despite the implied formalism in the data definition as discussed above, the pervasiveness of this representation on the web may be immediately obvious to the reader. Nevertheless, Figure 2.1 presents a collage of attribute based representations for different kinds of objects in the web such as gadgets, sports personalities and nations.

## 2.2 Attribute Types

The set of attributes that make up the schema could vary in the nature of values that they take. For example, the *age* attribute in a people database would typically take numeric values, whereas the *name* attribute would take short strings as values.



D.P and P.M. Deshpande, *Operators for Similarity Search*,
SpringerBriefs in Computer Science, DOI 10.1007/978-3-319-21257-9_2


Fig. 2.1: Collage of Attribute-based Representation Screenshots

We will now take a brief look at the types of attributes that are commonly used in databases upon which similarity search systems work.

- **Numeric:** Readings from sensors such as those that measure climatic attributes such as temperature and pressure would naturally take on numeric values. The popular Iris dataset[1] of flowers models flowers in terms of their attributes such as *sepal* and *petal* lengths and widths, all of which are numeric attributes. The population attribute for a countries dataset would also take numeric values. While all of the above take single numeric values, there could be attributes that take multi-numeric values such as locations in a x-y co-ordinate space, where a pair of numeric attributes for a 2-tuple represent each location. Though not strictly numeric, the latitude-longitude representation of geographic locations are also similar in the sense that both come from a well-ordered set.
- **Categorical:** Categorical attributes are another popular type, which can take on one of a limited number of possible values. The *gender* attribute is a common example of a categorical attribute. Other attributes such as *education* that can take on values such as *undergraduate*, *graduate* etc. are also categorical attributes. For a database of computing servers, the *operating system* would be a categorical attribute.
- **Text:** People profiles could contain attributes such as *short bio* or *address* fields that are typically textual where similarity assesment could be done between values using text similarity measures.
- **Sequence/Time Series:** Genes and proteins are typically attributes that are represented using a sequence of bases or amino acids respectively. Other common examples of sequences include health information such as ECG or EEG. Data from any kind of sensors are also naturally time series data.

[1] http://en.wikipedia.org/wiki/Iris_flower_data_set - Accessed May $^{th}$, 2015

The above list of kinds of attribute data are merely meant to be illustrative of the variety in attribute types. As is probably obvious, this list is not meant to be comprehensive.

## 2.3 Comparing Attribute Values: Similarity and Distance

Computing the similarity between two values of an attribute is a fundamental operation in similarity search systems. As seen in the introduction chapter, attribute-wise similarities are then aggregated in a manner as determined by the similarity operator, to arrive at a pair-wise similarity measure for objects. Though glossed over earlier, it is useful to point out the inter-convertibility between the notions of similarity and distance in typical scenarios. As is evident from the name, similarity and distance (i.e., dissimilarity) are inversely related; two values that have a large distance between them are likely to be judged to score low on similarity and vice versa. Similarity measures are generally expected to be in the range $[0,1]$ whereas many popular distance measures estimate distances to be any non-negative value. In the special case where distance is computed to be in the range $[0,1]$, the conversion could be simply done by subtracting the distance from unity[2]:

$$s = 1 - d \tag{2.1}$$

where $s$ and $d$ refer to similarity and distance respectively. For strictly positive distances, the reciprocal would serve as a measure of similarity. i.e.,

$$s = \frac{1}{d} \tag{2.2}$$

For a general non-negative distance measure, division by zero could be avoided by adding unity to the denominator as follows:

$$s = \frac{1}{d+1} \tag{2.3}$$

Another possible transformation [4] that was alluded to in Chapter 1 uses exponentiation:

$$s = e^{-d} \tag{2.4}$$

One may want to *control* this transformation in such a way that similarity be estimated to be 0.5 (50% of maximum similarity) at a particular value of distance. This could be done by using another parameter $\tau$ in the transformation such as below:

$$s = e^{\frac{-d}{\tau}} \tag{2.5}$$

[2] http://people.revoledu.com/kardi/tutorial/Similarity/WhatIsSimilarity.html - Accessed January 20, 2015

Under this transformation, $s$ would get estimated to 0.5 when the distance is $0.693 \times \tau$. Instead of choosing a pivot for 50% of the maximum similarity, it is straightforward to extend this to other pivots. A combination of transformations could also be used to convert distance to similarity; for example, one may use the sigmoid function[3] to transform the general non-negative distance measure to restrict it to the $[0,1]$ range, and then use the subtraction transformation as in Equation 2.1.

In short, there are a variety of possible mechanisms to convert similarity to distance and vice versa. In this book, when we consider comparing values, we will describe distance or similarity based on what is more convenient to describe; for example, distance is a very natural measure to compare numbers, whereas vectors may be compared by measuring their cosine similarity. It would be left to the reader to do the transformation between distance and similarity as needed, such as for implementation purposes.

## 2.4 Similarity Measures

A similarity measure, or a similarity function, is an attribute-specific function that quantifies the similarity between two values of the attribute. In this section, we will briefly look at a few common similarity measures, for various kinds of attributes.

- **Numeric Attributes:** For simple numeric attributes such as *age*, the difference between two values could be used as a measure of distance/dissimilarity. For multi-numeric attributes such as a pair of $(x,y)$ co-ordinates or a vector of values, the Minkowski distance metric is often used as a measure of distance. The Minkowski distance[4] between two vectors $v_1$ and $v_2$ would be estimated as:

$$\left( \sum_i (|v_1[i] - v_2[i]|)^p \right)^{1/p} \tag{2.6}$$

  For $p = 1$, this is equivalent to the sum of the differences between the corresponding values (i.e., the Manhattan distance[5], whereas $p = 2$ yields the Euclidean distance [1]). When $p \to \infty$, the Minkowski distance is simply equal the to maximum of the element-wise distances.
- **String Attributes:** Two values of a string attribute that are equal in length could be compared using the Hamming distance [2], that simply counts the number of positions in which the strings have different characters. For strings that need not necessarily be equal in length, Levenshtein distance [3] is often used. The Levenshtein metric considers a set of three operations, deletion, insertion and substitution, and counts the minimum number of operations that are needed to transform one string to the other. The Levenshtein distance between two strings,

[3] http://en.wikipedia.org/wiki/Sigmoid_function - Accessed 21$^{st}$ January, 2015

[4] http://en.wikipedia.org/wiki/Minkowski_distance - Accessed 22$^{nd}$ January, 2015

[5] http://mathworld.wolfram.com/ManhattanDistance.html - Accessed 22$^{nd}$ January, 2015

$a$ and $b$ is estimated as $lev_{a,b}(|a|,|b|)$ that uses a recursive function as follows:

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & \text{if } min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+I(a[i] \neq b[j]) \end{cases} & \text{otherwise} \end{cases} \tag{2.7}$$

where $I(.)$ is the indicator function that evaluates to 1 and 0 when the inner condition is true and false respectively. The Damerau-Levenshtein distance[6] extends the Levenshtein distance function by allowing for an additional operation, that of transposition of adjacent characters. Consider two strings *rick* and *irck*; these have Hamming and Levenshtein distance both evaluating to 2.0 due to mismatches on the first two characters, whereas these could be made identical by just one transposition leading to them being judged as just a unit distance away from each other according to the Damerau-Levenshtein distance.

- **Text Attributes:** Text attributes are typically represented as bags of words, wherein the ordering of the words in the text data is discarded. For example, the text fragment *this is a dog* would be represented as $\{this = 1.0, is = 1.0, a = 1.0, dog = 1.0\}$; the value associated with each word denotes its frequency in the text fragment. Each text fragment could be thought of as a vector in a (highly) multi-dimensional space where each dimension corresponds to a word in the dictionary, with the value on each dimension being equal to the frequency of the corresponding word in the text fragment. Thus, our sample text fragment takes on non-zero values only on the four dimensions $\{this, is, a, dog\}$. Instead of using simply the word frequency, the popular tf-idf weighting scheme[7] uses the product of word frequency and the IDF, a measure inversely related to the popularity of the word in a collection of documents such as news articles, to associate each dimension with. Whatever be the vector representation, word frequency based or tf-idf based, two text fragments are generally compared using the cosine similarity measure[8] that quantifies the cosine of the angle between their vector representations in the multi-dimensional space. Two identical vectors would have $0^o$ between them, and would thus be evaluated to have a similarity of unity.
- **Geographic Locations:** Geo-locations such as latitude-longitude pairs form another kind of popular attribute type. The distance between two points on the earth's surface is often estimated using the great circle distance, the shortest distance between two points along the great circle[9] that contains both the points. The Haversine formula[10] is a popular method to calculate the great circle distance.
- **Time-Series Data:** ECG, EEG and other kinds of temporally sequenced information form an integral part of personal health profiles; thus, similarity search for

[6] http://en.wikipedia.org/wiki/Damerau-Levenshtein_distance - Accessed January $22^{nd}$, 2015

[7] http://en.wikipedia.org/wiki/tf-idf - Accessed January $28^{th}$, 2015

[8] http://en.wikipedia.org/wiki/Cosine_similarity - Accessed January $28^{th}$, 2015

[9] http://en.wikipedia.org/wiki/Great_circle - Accessed January $28^{th}$, 2015

[10] http://mathforum.org/library/drmath/view/51879.html - Accessed January $28^{th}$, 2015

diagnostic services would need to estimate similarity between time-series data. Time-series data also are common in climate-related use cases where environmental factors such as temperature and humidity are measured over elongated periods of time. A simple method of comparing two time-series data would be to decide on a time interval - say, 5 seconds - and compare the values of the time series pair after the lapse of each interval. Thus, the values of the first time series at 5 seconds and 10 seconds from the start would be compared with the values of the second time series at 5 seconds and 10 seconds respectively. These pairwise comparisons would then be aggregated to arrive at a single similarity value for the time series pair. Though simple and appealing, this fixed interval comparison approach has several drawbacks, of which we will descibe one using an example. More often than not, there are periodicities in time-series data (e.g., ECG), and there are bound to be slight differences in the time intervals of these periodic patterns across time series; the fixed interval comparison method, due to usage of fixed intervals, would get out of sync and compare the crest of one time series against the trough of the other every now and then, resulting in exaggerated distance estimates. Dynamic Time Warping [5], among the more popular methods to compare time-series data, is robust to speed variations and uses a simple recursive formulation to compute a more realistic distance estimate.

## 2.5 Summary

In this chapter, we introduced the attribute-based object representation that is widely used for representing real-world entities on the web as well as in similarity search systems. This was followed by a brief description of various attribute types as well as measures used to quantify similarity and dissimilarity for specific attribute types. It was also noted that similarity and dissimilarity (distance) are often interchangeable through simple transformations, and that once either of them is specified, it is easy to derive the other.

## References

1. M. M. Deza and E. Deza. *Encyclopedia of distances*. Springer, 2009.
2. R. W. Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.
3. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707, 1966.
4. R. N. Shepard. Toward a universal law of generalization for psychological science. *Science*, 237(4820):1317–1323, 1987.
5. X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309, 2013.

http://www.springer.com/978-3-319-21256-2

Operators for Similarity Search
Semantics, Techniques and Usage Scenarios
P, D.; Deshpande, P.M.
2015, XI, 115 p. 44 illus., Softcover
ISBN: 978-3-319-21256-2