The relationship between HRR-based similarity and similarity based on structural kernels

Tony A. Plate

September 30, 2003

This is an extract from Chapter 6 in the book "Holographic Reduced Representation: Distributed Representation for Cognitive Structures", Tony A. Plate, CSLI Publications, 2003. This extract examines the relationship between two different ways of computing the similarity of structures: vector dot product similarity of HRRs and similarity based on structural kernels.

1 Relationship between HRR similarity and kernels for discrete structures

Work in machine learning on kernel-based methods over discrete structures, such as strings and trees, uses a variety of *kernels* to measure similarity between structures (Haussler, 1999, Collins and Duffy, 2002, Bod, 1998). For example, a kernel for strings could count the number of matching substrings, and kernel for trees could count the number of matching subtrees. A kernel is always a dot product between two feature vectors, i.e., a function K where $K(x, y) = \mathbf{h}(x) \cdot \mathbf{h}(y) = \sum_i h_i(x)h_i(y)$. The function $\mathbf{h}$ is some mapping of structures onto numerical vectors – $h_i(x)$ is the value of the ith feature of structure x. For example, in a kernel for strings, $h_i(x)$ could be the number of times the ith substring (in some enumeration of all substrings in the data set of interest) occurs in string x. What makes kernel-based methods practical is that there are efficient methods for computing $K(x, y)$ that do not require $\mathbf{h}(x)$ (or $\mathbf{h}(y)$) to ever be explicitly represented or even enumerated. This is fortunate, as the length of $\mathbf{h}$-vectors can be exponential in the size of the structures being represented – even the number of non-zero elements in $\mathbf{h}(x)$ can be exponential in the size of x. These efficient algorithms are typically based on dynamic-programming techniques and calculate $K(x, y)$ in time polynomial in the size of x and y. For some kernels there are even algorithms with quadratic time and linear time complexity (e.g., Vishwanathan and Smola, 2003, Lodhi et al., 2000).

Kernel-based machine learning algorithms have shown some degree of success in challenging tasks, e.g., text classification (Lodhi et al., 2000), and natural language parsing (Collins and Duffy, 2002). Hence it is interesting to see how

HRR similarity compares with kernels. In a straightforward but not particularly interesting way, HRR similarity is a kernel because it is the inner product of two vectors. However, there is a deeper sense also in which the HRR similarity is a kernel (approximately). Recall that a HRR can be viewed as a normalized superposition of vectors and binding chains (Sections **??** and **??**) ("binding chains" are convolutions of two or more of vectors). For the purposes of this discussion it is easier if we ignore normalization, and use symbols like $\mathbf{x}_1$ to denote binding chain vectors. Thus we can express the HRR for structure X as the superposition of m_x vectors, i.e.,

$$\mathbf{X} = \mathbf{x}_1 + \mathbf{x}_2 + \ldots + \mathbf{x}_{m_x},$$

and the HRR for the structure Y as the superposition of m_y vectors, i.e.,

$$\mathbf{Y} = \mathbf{y}_1 + \mathbf{y}_2 + \ldots + \mathbf{y}_{m_y}.$$

Then, using basic algebra, the dot product $\mathbf{X} \cdot \mathbf{Y}$ can be expressed as the sum of all pairwise similarities between the vectors in the HRRs for X and Y:

$$\mathbf{X} \cdot \mathbf{Y} = \sum_{i=1}^{m_x} \sum_{j=1}^{m_y} \mathbf{x}_i \cdot \mathbf{y}_j.$$

This expression shows that the dot product of HRRs is like a kernel over the feature space of all possible vectors and binding chains in the domain[1] – a massively larger feature space than the elements of the HRR vector.

This connection can be seen more clearly if we (somewhat unrealistically) assume that the binding chains and vectors comprising $\mathbf{X}$ and $\mathbf{Y}$ are constructed by convolving distinct and independently chosen random vectors. Then $\mathrm{E}[\mathbf{x}_i \cdot \mathbf{y}_j] = 0$, unless $\mathbf{x}_i$ and $\mathbf{y}_j$ are the same vector or binding chain, in which case the expected value is 1. Further, assume that the dimensionality of the HRRs is sufficiently large that the variance of the dot products of binding chains can be ignored, so that we can just say that if $\mathbf{x}_i$ and $\mathbf{y}_j$ are different then $\mathbf{x}_i \cdot \mathbf{y}_j = 0$, and if $\mathbf{x}_i$ and $\mathbf{y}_j$ are the same then $\mathbf{x}_i \cdot \mathbf{y}_j = 1$. Then $\mathbf{h}$ is the feature vector of the kernel and $h_k(\mathbf{X}) = 1$ if $\mathbf{X}$ contains the kth binding chain, and $h_k(\mathbf{X}) = 0$ otherwise. Consequently we can write the HRR dot product as the kernel over $\mathbf{h}$:

$$\mathbf{X} \cdot \mathbf{Y} = \sum_{i=1}^{m_x} \sum_{j=1}^{m_y} \mathbf{x}_i \cdot \mathbf{x}_j \approx \sum_k h_k(\mathbf{X}) h_k(\mathbf{Y}).$$

As noted, this should be regarded as an approximate relationship, since variances will not be zero with any finite-dimensional HRR vectors.

In summary we can say the following initial statements about the relationship between HRR dot products and kernels:

[1] **A sufficient condition for the dot product of HRRs to be a kernel over this space is that $\mathbf{x}_i \cdot \mathbf{y}_j$ equals one if $\mathbf{x}$ and $\mathbf{y}$ are the same vector or binding chain, and zero otherwise.**

- The HRR dot product can be viewed as an approximate kernel over the feature space comprised of all possible vectors and binding chains. The vectors and binding chains are the appropriate level at which to take this view because they are independent features of the HRR.[2] (The randomizing property of convolution means that a convolution of independently chosen random vectors is effectively a new independently chosen random vector.)

- This approximate kernel can be computed in time linear in the dimensionality of the HRR vectors for a set of structures that are already encoded as HRRs. For a particular accuracy of approximation, the dimensionality of the HRR vectors should be proportional to the number of binding chains (conjunctive features) present in structures. Thus, for some specified accuracy of approximation, the time complexity of the HRR dot product is linear in the number of binding chains in the HRRs.

- Unlike with most existing kernels over discrete structures, in which the features are either identical or different, with HRRs the features themselves (i.e., the binding chains) can have continuous degrees of similarity. Another way of saying this is that with HRRs the feature comparison need not bottom out at a level of discrete objects where similarity is either 0 or 1.

- The use of normalization while constructing HRRs will affect which binding chains are more important in the dot product of HRRs.

- In the HRRs described in this book, the binding chains for tree structures are mostly subpaths of paths from the root of the tree to the leaves (contextualized HRRs do have other types of binding chains). It could be possible that there are other ways of constructing binding chains for tree structures that could improve the performance of HRRs for cognitive modeling or Artificial Intelligence purposes.

References

Bod, R. 1998. *Beyond Grammar: An Experience-Based Theory of Language*. No. 88 in Lecture notes. Stanford, CA: CSLI Publications, distributed by Cambridge University Press.

Collins, M. and N. Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 263–270.

[2] **This picture becomes more complicated when the vectors out of which HRRs are constructed are not independently chosen, but have some similarity structure, as in the examples in this book. I have not investigated the consequences of this.**

Haussler, D. 1999. Convolution kernels on discrete structures. Tech. Rep. UCS-CRL-99-10, UC Santa Cruz, Santa Cruz, CA. http://www.cse.ucsc.edu/haussler.

Lodhi, H., J. Shawe-Taylor, N. Cristianini, and C. J. C. H. Watkins. 2000. Text classification using string kernels. In *NIPS*, pages 563–569.

Plate, T. 2003. *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*. CSLI Publications.

Vishwanathan, S. V. N. and A. J. Smola. 2003. Fast kernels for string and tree matching. In S. Becker, S. Thrun, and K. Obermayer, eds., *Advances in Neural Information Processing Systems (15)*. Cambridge, MA: MIT Press.