Torchhd: An Open Source Python Library to Support Research on Hyperdimensional Computing and Vector Symbolic Architectures

Mike Heddes
mheddes@uci.edu
Igor Nunes
igord@uci.edu
Pere Vergés
pvergesb@uci.edu
Denis Kleyko
denis.kleyko@ri.se
Danny Abraham
dannya1@uci.edu
Tony Givargis
givargis@uci.edu
Alexandru Nicolau
nicolau@ics.uci.edu
Alexander Veidenbaum
alexv@ics.uci.edu

1 Department of Computer Science, University of California, Irvine, CA 92617, USA
2 Intelligent Systems Lab, Research Institutes of Sweden, 164 40 Kista, Sweden

Editor: Sebastian Schelter

Abstract

Hyperdimensional computing (HD), also known as vector symbolic architectures (VSA), is a framework for computing with distributed representations by exploiting properties of random high-dimensional vector spaces. The commitment of the scientific community to aggregate and disseminate research in this particularly multidisciplinary area has been fundamental for its advancement. Joining these efforts, we present Torchhd, a high-performance open source Python library for HD/VSA. Torchhd seeks to make HD/VSA more accessible and serves as an efficient foundation for further research and application development. The easy-to-use library builds on top of PyTorch and features state-of-the-art HD/VSA functionality, clear documentation, and implementation examples from well-known publications. Comparing publicly available code with their corresponding Torchhd implementation shows that experiments can run up to 100× faster. Torchhd is available at: https://github.com/hyperdimensional-computing/torchhd.

Keywords: hyperdimensional computing, vector symbolic architectures, distributed representations, machine learning, symbolic AI, Python library

1. Introduction

Hyperdimensional computing (HD) (Kanerva, 2009), also known as vector symbolic architectures (VSA) (Gayler, 2003), is a computing framework capable of forming compositional distributed representations. It originated within artificial intelligence and cognitive science (Smolensky, 1990; Kussul et al., 1991; Plate, 1991) from attempts to address limitations of earlier distributed representations (von der Malsburg, 1986; Fodor and Pylyshyn, 1988). HD/VSA forms a “concept space” by exploiting the geometry and algebra of high-
dimensional spaces. The central idea is to represent information with randomly generated vectors, called hypervectors. Together with a set of operations on these hypervectors, HD/VSA can represent compositional structures, which, in turn, enables features such as reasoning by analogy (Plate, 1994; Kanerva, 2000; Gayler and Levy, 2009; Rachkovskij and Slipchenko, 2012; Kleyko et al., 2015) and cognitive computing (Emruli et al., 2013; Rasmussen and Eliasmith, 2014; Hersche et al., 2023).

While the foundational ideas of HD/VSA have been around for some time, it was only recently that it started gaining momentum both theoretically (Frady et al., 2018; Thomas et al., 2021; Clarkson et al., 2023) and practically, attracting attention from the wider machine learning community (Graves et al., 2014; Rahimi et al., 2019; Neubert et al., 2019; Ganesan et al., 2021; Hersche et al., 2022). This interest is partially driven by the observation that the strengths of HD/VSA complement some of the limitations of current artificial neural networks (Greff et al., 2020; Smolensky et al., 2022), leading to hybrid approaches that could be viewed as neuro-symbolic (Hersche et al., 2023).

In face of the growing interest, there have been substantial efforts to consolidate and disseminate HD/VSA by its research community. The challenge for these initiatives has been the fact that HD/VSA involves many disciplines, including machine learning, neuroscience, electrical engineering, artificial intelligence, mathematics, and cognitive science. Keeping up with the latest advances is not trivial as publications are spread across different venues and disciplines. It is, therefore, important to pursue consolidation and integration efforts, which will facilitate further research endeavors. Despite detailed surveys of various aspects of the area (Ge and Parhi, 2020; Schlegel et al., 2022; Kleyko et al., 2022c, 2023), much less attention, however, has been paid towards streamlining the translation of concepts, methodologies, and algorithms into a coherent open source software.

To address this issue, we developed Torchhd—an open source library for HD/VSA. Up to date with the latest advances in the area, Torchhd aims to lower the barrier of entry to HD/VSA for novices and provides a high-performance execution platform for experienced researchers. To support a wide variety of HD/VSA primitives, applications, and research directions; the provided abstractions are designed to be modular, allowing users to adopt the aspects that support their workflow. Our design philosophy focuses first on ease-of-use, aiming to be accessible without limiting expressiveness. The ease of use allows for application-driven research to focus on the conceptual and methodological aspects of the project while, at the same time, Torchhd enables the study of individual elements of HD/VSA for theoretical research. The second goal of the library is to provide high-performance execution, enabling much faster evaluation of novel algorithms without hurting the Python idiomatic development experience. We believe that Torchhd will benefit the HD/VSA and the broader machine learning communities based on its following features: 1) state-of-the-art HD/VSA methods for transforming data into hypervector; 2) support for automatic differentiation for research on hybrid neuro-symbolic models; 3) Python idiomatic interface design for a flexible, developer-first experience; 4) high-performance execution, which means that applications can run orders of magnitude faster; 5) interfaces for data sets commonly used in the literature to evaluate and benchmark HD/VSA methods (129 data sets currently).
2. Torchhd

Torchhd builds on PyTorch (Paszke et al., 2019), a library for high-performance tensor computation. PyTorch and its ecosystem provide many machine learning utilities, support for various hardware accelerators, and built-in automatic differentiation (autodiff) (Paszke et al., 2017). Building on PyTorch enables the exchange of knowledge between HD/VSA and the wider machine learning community, and facilitates the cross-fertilization of ideas.

Torchhd supports six common HD/VSA models, namely binary spatter codes (BSC) (Kanerva, 1997), multiply-add-permute (MAP) (Gayler, 1998), holographic reduced representations (HRR), Fourier HRR (FHRR) (Plate, 1995), sparse block codes (SBC) (Laiho et al., 2015; Frady et al., 2023), and vector-derived transformation binding (VTB) (Gosmann and Eliasmith, 2019). These models differ in their specification of the high-dimensional space and their implementations of the fundamental operations: superposition and binding of hypervectors. Support for (F)HRR enabled the implementation of vector function architectures (Frady et al., 2021, 2022) which extends an approach for implementing large-scale kernel machines via random Fourier features (Rahimi and Recht, 2007). Our implementations are verified with extensive automatic unit testing. We also place emphasis on clear documentation, which is available at: https://torchhd.readthedocs.io

The library has six modules which provide the following functionality:

- **functional**: functions for generating hypervectors (Rachkovskij et al., 2005; Nunes et al., 2023); the operations on hypervectors (Schlegel et al., 2022); and a resonator network for factorizing hypervectors (Frady et al., 2020; Renner et al., 2022; Kleyko et al., 2022a; Langenegger et al., 2023).

- **embeddings**: classes for transforming scalars or feature vectors into hypervectors, ranging from simple hypervector lookups till similarity-preserving transformations of feature vectors (Kleyko et al., 2021; Thomas et al., 2021), some of which approximate well-known kernels (Rahimi and Recht, 2007; Frady et al., 2022).

- **models**: common classification models such as the centroid model with class prototypes and its various training algorithms (Rahimi et al., 2016a; Imani et al., 2017; Hernandez-Cane et al., 2021; Nunes et al., 2022), learning vector quantization (Diao et al., 2021), and regularized least squares (Kleyko et al., 2021).

- **memory**: methods for long-term storage and retrieval of hypervectors, such as sparse distributed memory (Kanerva, 1988), inspired by the cerebellum (Teeters et al., 2022), the (modern) Hopfield network (Hopfield, 1982; Krotov and Hopfield, 2016), and the attention mechanism (Vaswani et al., 2017; Ramsauer et al., 2020).

- **structures**: classes for HD/VSA data structures (Kleyko et al., 2022b) such as hash tables, graphs, binary trees, and finite state automata (Osipov et al., 2017; Yerxa et al., 2018; Heddes et al., 2022). This module facilitates the development of classical algorithms using HD/VSA.

- **datasets**: convenient access to 126 classification and 3 regression data sets commonly used in the literature. This includes a classification benchmark of UCI machine learning repository data sets (Dua and Graff, 2017) created by Fernández-Delgado et al. (2014). All data sets are interoperable with the PyTorch ecosystem.
We note that several other software for HD/VSA exist. Table 1 contrasts the features of Torchhd with those of OpenHD (Kang et al., 2022), HDTorch (Simon et al., 2022), and VSA Toolbox (Schlegel et al., 2022). As part of their review on HD/VSA models, Schlegel et al. (2022) implement a broader range of HD/VSA models, however, their VSA Toolbox misses many of the features that make Torchhd particularly useful for research and development of HD/VSA, see, for example, Table 1. Importantly, Torchhd is the first general software library for HD/VSA since the other software were developed for more specific purposes.

To demonstrate the performance of Torchhd, we implemented three classification tasks and compared the execution time to the original publicly available code. All the experiments ran on a machine with 20 Intel Xeon Silver 4114 CPUs, 93 GB of RAM, and 4 Nvidia TITAN Xp GPUs, only a single CPU or GPU was used. The results in Table 2 show that Torchhd is faster for all examples with an average of 24× and 54× faster for CPU and GPU, respectively. The provided abstractions ensure performant code by avoiding overly sequential execution, reinforced by our support for batch processing.

### Table 1: A qualitative assessment of the functionality of HD/VSA software.

<table>
<thead>
<tr>
<th>Library</th>
<th>BSC</th>
<th>MAP</th>
<th>HRR</th>
<th>FHRR</th>
<th>Memory models</th>
<th>Data structures</th>
<th>Resonator Network</th>
<th>Auto-diff</th>
<th>Data sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenHD</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>HDTorch</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>±</td>
<td>✗</td>
</tr>
<tr>
<td>VSA Toolbox</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Torchhd (ours)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

### Table 2: Execution time in seconds (and speedup)

<table>
<thead>
<tr>
<th>Paper</th>
<th>Original</th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>EU languages (Rahimi et al., 2016b)</td>
<td>13111.73</td>
<td>542.83 (24×)</td>
<td><strong>125.89</strong> (104×)</td>
</tr>
<tr>
<td>EMG gestures (Rahimi et al., 2016a)</td>
<td>1152.32</td>
<td><strong>28.44</strong> (41×)</td>
<td>28.95 (40×)</td>
</tr>
<tr>
<td>VoiceHD (Imani et al., 2017)</td>
<td>277.97</td>
<td>37.74 (7×)</td>
<td><strong>16.17</strong> (17×)</td>
</tr>
</tbody>
</table>

3. Conclusions and Future Work

Torchhd is a Python library for hyperdimensional computing (HD) a.k.a. vector symbolic architectures (VSA) that aims to be simple, versatile, and highly performant. The library builds on PyTorch, thus, enabling the exchange of knowledge between the HD/VSA and the broader machine learning communities as well as making PyTorch’s vast ecosystem of deep learning tools available for HD/VSA. Torchhd is designed to include any HD/VSA model and already supports binary spatter codes, multiply-add-permute, (Fourier) holographic reduced representations, sparse block codes, and vector-derived transformation binding. We plan to keep the library up-to-date with novel developments in the area by including more HD/VSA models and supporting a broader spectrum of learning algorithms, such as differentiable learning, regression, density estimation, and clustering. Our work joins the community’s vital effort to consolidate and disseminate HD/VSA research.
Acknowledgments

We would like to thank Ross W. Gayler for helpful discussions on the design choices and desired functionality of the library; Rishikanth Chandrasekaran, Dheyay Desai, Jenny Lee, and Xiaofan Yu for their contributions to the library; and finally, Anthony Thomas for his support while resolving an issue with the random projection implementation.

DK received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 839179.

References


