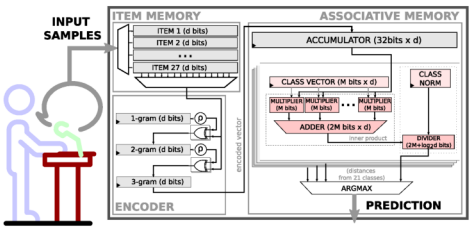
A probability-inspired normalization for fixed-precision Hyper-Dimensional Computing

Sohum Datta
Electrical Engineering and Computer Sciences
University of California Berkeley
Berkeley, CA, USA
sohumdatta@berkeley.edu

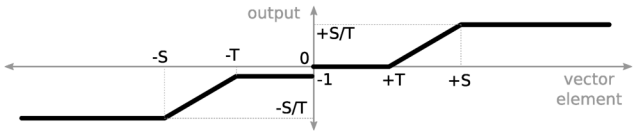
Jan M. Rabaey
Electrical Engineering and Computer Sciences
University of California Berkeley
Berkeley, CA, USA
jan_rabaey@berkeley.edu

***Abstract*—Hyper-Dimensional Computing (HDC), a promising nano-scalable paradigm for low-energy predictions and lightweight learned models, has seen a surge of interest from the hardware accelerator community. However, the classical single-bit per vector element approach for HDC seldom achieves higher classification accuracy than multi-bit alternatives, and is inadequate to support the rapidly growing application space. A great challenge for multi-bit HDC hardware is to negotiate the enormous increase in logic vis-á-vis the single-bit hardware. Key to minimizing this cost is to limit bits per vector element, which is potentially unbounded without transformation, and can be very large for some applications. This work proposes a hardware-friendly numerical transformation on a HDC vector where the result has fixed bits per element. Under a reasonable assumption on the vector's distribution, it is proven that the transformation guarantees at most a small, known error in associative search. Verification experiments indicate the theoretical guarantee is very pessimistic; the actual error is at most 18% of the upper bound. Estimates predict 3.8X hardware savings with a 0.04% accuracy drop. We believe emerging stochastic approaches offer an exciting new opportunity of employing high-dimensional probability theory for accelerator design.**

***Index Terms*—hyper-dimensional computing, energy efficiency, Internet-of-Things (IoT), probability, machine learning**



(a) A multi-bit HDC inference system for classification.



(b) The transfer function of proposed element-wise numerical transformation.

Fig. 1. **Overview.** The multi-bit HDC system is far more expensive in logic gates than the binary system for computing predictions. In (a), inner-product (dashed box) replaces bitwise XOR and popcount logic of binary HDC, and adds logic for integer division (dotted box). The proposed transformation (b) limits the logic overhead and is robust and essentially lossless in practise.

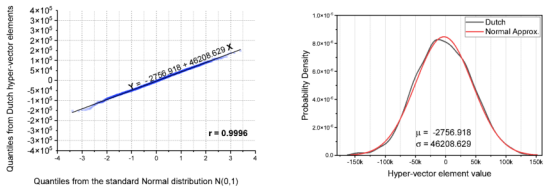
I. Introduction

The Hyper-Dimensional Computing (HDC) paradigm uses d-dimensional random vectors (d is in thousands) as the principal data-type. Conceptually, it captures the high-dimensional and random nature of neural representation in parts of the human brain, which can then be transformed using simple operations such as bitwise XOR and a permutation to perform symbolic reasoning [1], implement data structures [2], and supervised classification (reviewed in [3]). Although half a dozen HDC variants exist [4], the Multiply-Add-Permute model using $0, 1$ or $+1, -1$ as vector elements (henceforth, binary model) has been most widely adopted in the hardware community due to its hardware-friendliness ([5] provides a brief survey). Supervised classification has seen the most successful application of HDC so far, and will be the object of study in this work.

When using HDC for supervised classification, as illustrated by (figs. 4 and 5 of) [6], the binary model almost always has lower classification accuracy than the model using integers to represent elements of the trained class vector (henceforth, integer model). Furthermore, even though increasing vector dimension improves binary accuracy (fig. 7 of [5]), the integer model uniformly has greater or equal accuracy for every dimension [6]. This is because each element contains identical information as every other element, limiting the total information capacity [7]. Contrast this with the integer model, where bits in each element differ in significance, thus requiring a smaller vector dimension for the same information capacity. For new applications such as HDC factorization [8], intermediate representations require multiple bits/element.

Fig. 1 (a) shows main components of an integer model data path for performing predictions in a supervised classification task. Within the Encoder, ρ denotes the permutation operation and XOR gates denote the bitwise XOR or binding operation. Since only class vector elements change from binary to integer model, all hardware modifications to a binary HDC hardware (Fig. 3 of [5]) reside within the associative memory. The hamming distance $\mathrm{d_H}$ is no longer suitable as a distance metric for the integer model; the cosine similarity $\mathrm{d_{cos}}$, i.e. the inner-product of test vector with each class vector divided by the class vector norm, is applicable and a common choice [3].

This work was supported by TSMC and DARPA under the HyDDENN program. The authors would like to thank Bruno Olshausen, Denis Kleyko, Alisha Menon and Laura I. Galindez from UC Berkeley for their feedback.



(a) Normal probability plot (b) Distributions

Fig. 2. The normal distribution is a good fit for the Dutch hyper-vector.

However $\mathrm{d}_{\cos}$ is expensive to implement in hardware: M-bit adders and multipliers need $O(M \log M)$ and $O(M^2)$ complex gates respectively with $O(\log M)$ delay [9], and d of them (i.e. thousands) are required for each class vector. A register and divider are needed to store the class norm and divide the inner product for each class. Therefore, the only possible approach to guarantee feasibility of any multi-bit HDC hardware must be to reduce or (ideally) fix the required bits/element M. M can be arbitrarily large as it is determined by the number of terms superimposed [7] which is usually the number of class examples used in training. Thus, a large dataset like [10] would require $M \geq 20$ bits/element of `signed int`.

For a principled procedure to fix M, observe the following about cosine similarity (denoted $\mathrm{d}_{\cos}$) with a class vector:

- Class vector elements with relatively large magnitude have a great effect on cosine similarity. But large magnitudes may be very unlikely, like the application considered here.
- Conversely, magnitude of a class vector element is likely to be small, but negligibly affects cosine similarity.

Thus, the idea is to precisely balance vector-element magnitude with its probability. To obtain theoretical results (see Appendix), the following assumption formalizes the above idea:

Normality Assumption. The class hyper-vector elements are independent and identically distributed, and follow a (zero mean) normal distribution. ■

The i.i.d. nature of elements are implicit as item vectors are generated as i.i.d. and HDC operations preserve it. Shannon's seminal work [11] argues that for classification, languages can be adequately modeled as *independent* samples from the empirical n-gram distribution. Independence is essential because when HDC is used for language recognition [10], the superimposed n-gram vectors of a language are i.i.d. bipolar; thus the final elements can be accurately approximated as a 0-mean normal distribution. Indeed, for EUROPARL data set used in [10], fig. 2 and table I show that the empirical distribution agrees with the normality assumption for each language hyper-vector. In particular, for at least 95% of samples (i.e. $\mathcal{N}(0,1)$ is within 2 std. devs. of mean 0) in all languages, a line is a great fit in the normal probability plot (table I). Since $\mu/\sigma < 0.06$, the mean is essentially zero. Therefore, there is little evidence to reject the normality assumption for EUROPARL. HDC language classification using EUROPARL is used to verify results about the proposed transformations.

II. Analysis

The normality assumption is henceforth presumed valid. Before applying numerical transformations, trained class hyper-vectors must be held in `ACCUMULATOR` (fig. 1(a)). To allow training large datasets, `ACCUMULATOR` must have large bits/element (eg. 32 bits/element as shown), for both binary and multi-bit hardware. For transformations, additional logic to calculate the std. dev. across elements denoted by σ (table I) is required. However, this calculation is done only once: after training and before the numerical transformations. It is assumed that $d = 2048$ since this is sufficient for most applications [5].

The first objective is to limit the maximum magnitude of the class hyper-vector elements. For a positive (integer) S:

Saturation. Replace element with S if value is larger than S, replace element with $-S$ if smaller than $-S$; keep unchanged otherwise. ■

Corollary 1 shows how the probability of *no change* to the vector after saturation is determined by S. There is a 99% chance of no change when saturating at $S \geq 1.05 \times \sigma\sqrt{2048}$, and all EUROPARL class vectors remain unchanged (fig. 3(a)). As $S \downarrow 0$, accuracy decreases as more elements are saturated.

For the next transformation, choose a positive (integer) T:

Normalization. `int` division of each element by T. ■

A crucial consequence of normalization is elements with magnitude $< T$ become 0. Zeroing out small magnitude elements (henceforth, thresholding) in a class hyper-vector was proposed to increase sparsity [12]. While [12] reports empirical advantages, they do not analyze the mechanism of this improvement. For magnitudes $\geq T$, effects of zeroing out quotients' fractional part due to `int` division may be neglected to a first order because usually $S >> T$.

Theorem 1 provides a recipe for T that guarantees error in $\mathrm{d}_{\cos}$ is at most a chosen ϵ with $1 - \delta = 0.99$ probability. For a generally useful bound, a reasonable choice would be $\epsilon \leq \sqrt{2/\pi}$, the cosine similarity between a normal vector and its bipolarized version in high dimensions [13]. This is because after normalization, one should (at least) be able distinguish the class vector from its bipolarized version used in the binary model. Using $\epsilon \leq \sqrt{2/\pi}, \delta = 0.01, d = 2048$ gives $T \leq 0.3\sigma$. Fig. 3(c) shows the maximum absolute error introduced in the cosine similarity of each class vector and all test vectors of EUROPARL due to the normalization transform; Dutch exhibits the largest absolute error introduced but is consistently $< 0.18\epsilon$. Normalization with $T = 0.3\sigma$ gives 20360 correct of 21000 tests (96.95% accuracy), compared to 20370 correct for the untransformed cosine (97.00% accuracy).

Putting it all together: saturation followed by normalization (Corollary 2) can be used to set M to be $\left\lceil \log_2(1.1\pi\sqrt{d}) \right\rceil + 1$ bits (extra bit is the sign bit in 2's complement representation) and guarantee at most $\sqrt{2/\pi}$ error in $\mathrm{d}_{\cos}$ with 99% chance. For $d = 2048$, this leads to $M = 9$.

The inference data path of fig. 1(a) can be optimized further. In particular, division by norm of the (transformed) class vector (dotted box in fig. 1(a)) is no longer needed as normalization by $T \propto \sigma = \sqrt{(1/d)\sum_{i=1}^{d}(x_i^2 - \mu)}$ already achieves this ($\mu \approx 0$). Indeed, in fig. 3(b) the "normalization & $\mathrm{d}_{\cos}$" (symbol: ■) and "normalization & inner-product" (symbol: •) plots have almost equal accuracy until $T \leq 1.5\sigma$: they diverge beyond as most elements ($> 90\%$) are thresholded to 0. For $T = 0.3\sigma$, normalization & inner-product gives 20358 correct.

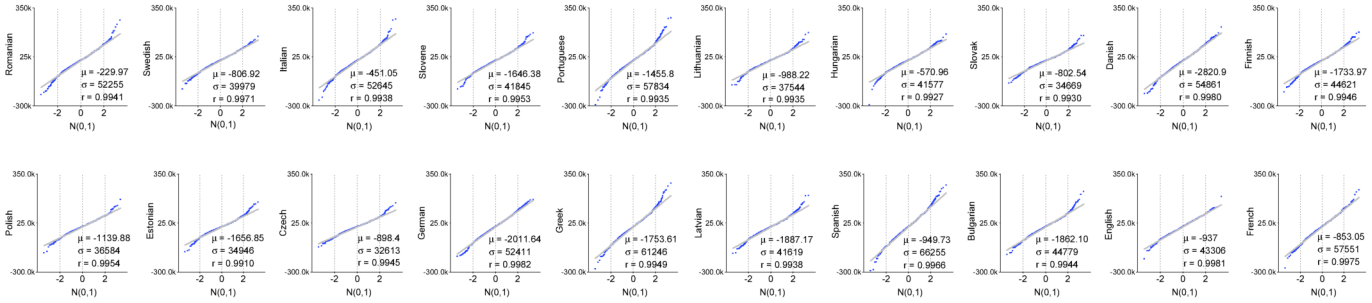
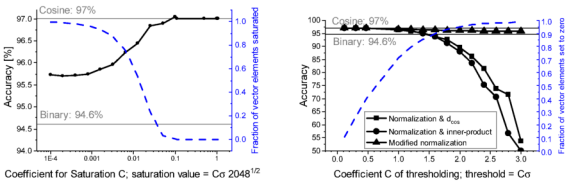


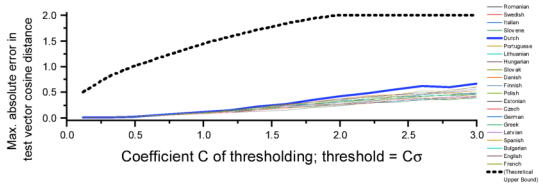
TABLE I

Normal Probability Plots. The quantile-quantile plots for languages other than Dutch in EUROPARL dataset ($d = 2048$) and the best fitting lines. Language vector elements are modeled as $Y = \mu + \sigma X$ where $X \sim \mathcal{N}(0, 1)$. The coefficient of determination $0 \leq r \leq 1$ is specified; $r = 1$ indicates perfect modeling.



(a) Saturation & accuracy. Right axis used by dashed plot.

(b) Normalization & accuracy. Right axis used by dashed plot.



(c) Errors induced in d_{cos} by Normalization

Fig. 3. Proposed transformations are almost lossless in classification accuracy.

A final avenue for optimization is to correct the *dramatic collapse in accuracy* after normalization as T increases (beyond theory-required $T \leq 0.3\sigma$). To that end, a modified saturation and normalization scheme is proposed.

Modified Saturation and Normalization. For positive integers $S >> T$.

1) Saturate the class by S.
2) Left shift each element in `ACCUMULATOR` by $R = 31 - \lceil \log_2 S \rceil$ bits
3) `int` divide each element by T.
4) Arithmetic right shift the quotient in `ACCUMULATOR` by $32 - M$ bits, so that sign bit is in position $M = \lceil \log_2(S/T) \rceil + 1$
5) Compute inner-product of test vector with all transformed class vectors.
6) The class with largest inner product is prediction. ■

This prevents accuracy collapse for large T by using distinction: very small negative numbers become -1 but positive numbers become 0 (fig. 1(b)). For large T, the transformation is essentially just bipolarization, and bipolarization approximately preserves direction of a normal vector [13]. Thus, if there aren't enough final bits/element, the accuracy of modified normalization should approach binary accuracy. Indeed, fig. 3(b) shows the "Modified normalization" accuracy plot (symbol: ▲) degrades gracefully as T increases, approaching the binary accuracy for HDC language classification.

An early estimation of the hardware cost in terms of logic gates is tabulated in table II. All Flip-Flop bits (FFs) from the inference data path (fig. 1(a)) are included. Multipliers use a

HDC Model	FF (bits)	HAs	FAs	CLBs	# Transistors
(A) Orig. Cosine	934,402	3,515,925	18,051,411	15,118,425	840,857,808
(B) Final Cosine	458,752	1,590,645	3,827,103	4,857,405	223,223,784
(A) / (B)	2.04	2.21	4.72	3.11	3.77

TABLE II

Logic cost of prediction. $d = 2048$. Half-Adders (HAs), Full-Adders (FAs), Carry-Lookahead Blocks (CLBs) and total CMOS "transistor cost" for d_{cos}.

Wallace-tree reduction before addition and adders use Kogge-Stone logic with Carry Lookahead Blocks (CLBs) [9]. The cost of "Original Cosine" (row A) divider is also included. The total CMOS transistor cost of all half-adders (18 each), full-adders (28 each) and CLBs (18 each) is a preliminary estimate of energy expenditure for a HDC inference.

As shown in table II, a $2 - 4\times$ improvement is achieved in all aspects of estimated hardware complexity. The CMOS transistor cost predict an overall saving of $3.77\times$ due to the proposed transformations for HDC language classification.

III. Conclusions

An element-wise magnitude $2^{M-1} \approx \pi\sqrt{d}$ is sufficient for (conservatively) limiting the error in cosine similarity to a reasonable upper bound. This allows reducing $M = 20$ (97.00% accuracy) to $M = 9$ (96.96% accuracy) and results in $\approx 3.8\times$ lower combined logic cost for HDC language classification. Finally, the modified transformation allows a graceful degradation to the binary accuracy as $M \downarrow 1$.

The theoretical tool employed here is the tail behavior of χ^2_d random variable. A future work could generalize this result to other distributions for other applications, such as truncated Normal, Beta and Pareto as well as members of Sub-Gaussian, Sub-Gamma and Exponential families.

Appendix

The normal distribution with mean μ and standard deviation $\sigma > 0$ is denoted by $\mathcal{N}(\mu, \sigma^2)$. Given independent random variables $x_1, x_2, \ldots, x_d \sim \mathcal{N}(0, 1)$, the squared euclidean norm of the vector $X = (x_1, x_2, \ldots, x_d)$ given by $\|X\|^2 = \sum_{i=1}^{d} x_i^2$ obeys the Chi-squared distribution with d-degrees, denoted χ^2_d.

Lemma 1 (χ^2_d concentration). If $Z \sim \chi^2_d$ then $\Pr[\frac{Z}{d} - 1 \geq t] \leq e^{-dt^2/8}$ and $\Pr[\frac{Z}{d} - 1 \leq -t] \leq e^{-dt^2/8}$ whenever $0 < t < 1$. (See example 2.11 of [14] for an elementary proof.)

Corollary 1 (Saturation). Let $X = (x_1, x_2, \ldots, x_d)$ where $x_i \sim \mathcal{N}(0, \sigma^2)$ are independent. Given $C > 0$, define X_C to be the vector obtained from X whose i^{th} element is C if $x_i > C$, $-C$ if $x_i < -C$, and x_i otherwise. If $e^{-d/8} < \delta < 1$ then $C \geq \sigma\sqrt{d + \sqrt{8d \log 1/\delta}}$ implies $\Pr[X_C = X] \geq 1 - \delta$. (Note that the condition $e^{-d/8} < \delta < 1$ is not really constraining δ: $d \geq 500 \implies e^{-d/8} < 7.2 \times 10^{-28}$.)

Proof. See that $\max_{1 \leq i \leq d} |x_i| \leq \|X\|$. Thus, $\Pr[X_c = X] = \Pr[\max_i |x_i| \leq C] \geq \Pr[\|X\| \leq C]$.
Now $\|X\|^2/\sigma^2 \sim \chi^2_d$. From Lemma 1, $\Pr[\|X\|^2/\sigma^2 \geq d(1+t)] \leq e^{-dt^2/8}$ whenever $0 < t < 1$. Subsituting t from $e^{-dt^2/8} \leq \delta$ gives $d(1+t) \geq d + \sqrt{8d \log 1/\delta}$ which is valid provided $\delta > e^{-d/8}$. Hence we have $\Pr\left[\|X\| \leq \sigma\sqrt{d + \sqrt{8d \log 1/\delta}}\right] \geq 1 - \delta$. Finally, if $C \geq \sigma\sqrt{d + \sqrt{8d \log 1/\delta}}$ then $\Pr[X_C = X] = \Pr[\max_i |x_i| \leq C] \geq \Pr\left[\|X\| \leq C\right] \geq 1 - \delta$ ■

Theorem 1 (Thresholding). Let $X = (x_1, x_2, \ldots, x_d)$ where $x_i \sim \mathcal{N}(0, \sigma^2)$ are independent. Given $C > 0$, let $\tau_C(X)$ be the vector obtained by applying thresholding function $\tau_C(\cdot)$ element-wise on X, so that the vector's i^{th} element is 0 if $|x_i| < C$ and x_i otherwise. Then for any vector $u \in \mathbb{R}^d$, error $0 < \epsilon \leq 2$ and $e^{-d/8} < \delta < 1$, we have $\Pr[|d_{\cos}(u, X) - d_{\cos}(u, \tau_C(X))| \leq \epsilon] \geq 1 - \delta$ whenever $C \leq \frac{\sigma}{2}\epsilon^2\sqrt{1 - \sqrt{(8/d)\log 1/\delta}}$

Proof. Inner product of vectors X, Y is written $X \cdot Y$. $\mathbf{1}_{\text{condition}}$ is indicator for 'condition', (1 when true, 0 otherwise).

From Lemma 1, $\Pr\left[\|X\| \geq \sigma\sqrt{d - \sqrt{8d \log 1/\delta}}\right] \geq 1 - \delta$ whenever $e^{-d/8} < \delta < 1$. Then

$$
\begin{aligned}
&|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, \tau_C(X))| \\
&= |(u/\|u\|) \cdot (X/\|X\|) - (u/\|u\|) \cdot (\tau_C(X)/\|\tau_C(X)\|)| \\
&= |(u/\|u\|) \cdot (X/\|X\| - \tau_C(X)/\|\tau_C(X)\|)| \\
&\leq \|(X/\|X\|) - (\tau_C(X)/\|\tau_C(X)\|)\| \text{ (Cauchy-Schwarz ineq.)} \\
&= \sqrt{2 - 2X \cdot \tau_C(X)/(\|X\|\|\tau_C(X)\|)} \\
&= \sqrt{2\Big(1 - (\sum_{i=1}^{d} x_i^2 \mathbf{1}_{|x_i| \geq C})/(\|X\|\|\tau_C(X)\|)\Big)} \text{ (defn. of } \tau_C(\cdot)) \\
&= \sqrt{2\big(1 - \|\tau_C(X)\|^2/(\|X\|\|\tau_C(X)\|)\big)} = \sqrt{2 - 2\|\tau_C(X)\|)/\|X\|} \\
&\leq \sqrt{2\|X - \tau_C(X)\|/\|X\|} \text{ (} \triangle \text{ inequality)} \\
&= \sqrt{2\frac{\sqrt{\sum_{i=1}^{d} x_i^2 \mathbf{1}_{|x_i| < C}}}{\|X\|}} < \sqrt{2\frac{\sqrt{\sum_{i=1}^{d} C^2}}{\|X\|}} < \sqrt{2C\frac{\sqrt{d}}{\|X\|}} \\
&\leq \sqrt{\frac{2C}{\sigma\sqrt{1 - \sqrt{(8/d)\log 1/\delta}}}} \text{ with prob. } \geq 1 - \delta \text{ (Lemma 1)}
\end{aligned}
$$

Using $C \leq (\sigma/2)\epsilon^2\sqrt{1 - \sqrt{(8/d)\log(1/\delta)}}$ gives the result. ■

Corollary 2 (Saturation followed by Normalization). Let $X = (x_1, x_2, \ldots, x_d)$ where $x_i \sim \mathcal{N}(0, \sigma^2)$ are independent. Given $0 < \epsilon \leq 2$ and $2e^{-d/8} < \delta < 1$ and any vector $u \in \mathbb{R}^d$, let $a = \sigma\sqrt{d + \sqrt{8d \log(2/\delta)}}$ and $b = \frac{\sigma}{2}\epsilon^2\sqrt{1 - \sqrt{(8/d)\log(2/\delta)}}$. Then the vector $Y = \frac{1}{b}\tau_b(X_a)$ satisfies $\max_{1 \leq i \leq d} |y_i| \leq 2\frac{\sqrt{d}}{\epsilon^2}\sqrt{\frac{1 + \sqrt{(8/d)\log(2/\delta)}}{1 - \sqrt{(8/d)\log(2/\delta)}}}$ and $\Pr[|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, Y)| \leq \epsilon] > 1 - \delta$

Proof. For the first part, note that the maximum absolute value of elements of X_a is a, which remains unchanged by $\tau_b(\cdot)$. Thus, the maximum absolute value of $\frac{1}{b}\tau_b(X_a)$ is $\frac{a}{b}$.

For the second part, begin from $\operatorname{d_{cos}}(u, Y) = \operatorname{d_{cos}}(u, bY)$ and use Corollary 1 and Theorem 1 to simplify:

$$
\begin{aligned}
&\Pr[|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, Y)| \leq \epsilon] \\
&= \Pr[|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, \tau_b(X_a))| \leq \epsilon] \\
&\geq \Pr[|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, \tau_b(X_a))| \leq \epsilon | X_a = X] \Pr[X_a = X] \\
&\geq \Pr[|\operatorname{d_{cos}}(u, X) - \operatorname{d_{cos}}(u, \tau_b(X))| \leq \epsilon](1 - \delta/2) \\
&\geq (1 - \delta/2)^2 > 1 - \delta \text{ ■}
\end{aligned}
$$

REFERENCES

[1] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, no. 2, pp. 139–159, 2009.

[2] D. Kleyko, M. Davies, E. P. Frady, P. Kanerva, S. J. Kent, B. A. Olshausen, E. Osipov, J. M. Rabaey, D. A. Rachkovskij, A. Rahimi, and F. T. Sommer, "Vector symbolic architectures as a computing framework for nanoscale hardware," *arXiv preprint arXiv:2106.05268*, 2021.

[3] L. Ge and K. K. Parhi, "Classification using hyperdimensional computing: A review," *IEEE Circuits and Systems Magazine*, vol. 20, no. 2, pp. 30–47, 2020.

[4] A. Rahimi, S. Datta, D. Kleyko, E. P. Frady, B. Olshausen, P. Kanerva, and J. M. Rabaey, "High-dimensional computing as a nanoscalable paradigm," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2508–2521, 2017.

[5] S. Datta, R. A. Antonio, A. R. Ison, and J. M. Rabaey, "A programmable hyper-dimensional processor architecture for human-centric iot," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 3, pp. 439–452, 2019.

[6] A. Hernández-Cano, C. Zhuo, X. Yin, and M. Imani, "Real-time and robust hyperdimensional classification," in *Proceedings of the 2021 on Great Lakes Symposium on VLSI*, 2021, pp. 397–402.

[7] E. P. Frady, D. Kleyko, and F. T. Sommer, "A theory of sequence indexing and working memory in recurrent neural networks," *Neural Computation*, vol. 30, no. 6, pp. 1449–1513, 2018.

[8] S. J. Kent, E. P. Frady, F. T. Sommer, and B. A. Olshausen, "Resonator Networks, 2: Factorization Performance and Capacity Compared to Optimization-Based Methods," *Neural Computation*, vol. 32, no. 12, pp. 2332–2388, 12 2020. [Online]. Available: https://doi.org/10.1162/neco_a_01329

[9] J. M. Rabaey, A. Chandrakasan, and B. Nikolić, "Designing arithmetic building blocks," in *Digital integrated circuits: a design perspective*, 2nd ed. Pearson Education, Inc., 2003, ch. 11, pp. 578–589.

[10] A. Joshi, J. T. Halseth, and P. Kanerva, "Language geometry using random indexing," in *International Symposium on Quantum Interaction*. Springer, 2016, pp. 265–274.

[11] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[12] M. Imani, S. Salamat, B. Khaleghi, M. Samragh, F. Koushanfar, and T. Rosing, "Sparsehd: Algorithm-hardware co-optimization for efficient high-dimensional computing," in *2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2019, pp. 190–198.

[13] A. G. Anderson and C. P. Berg, "The high-dimensional geometry of binary neural networks," in *International Conference on Learning Representations*, 2018.

[14] M. J. Wainwright, *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press, 2019, vol. 48.