Bayesian-Informed Hyperdimensional Learning for Intelligent and Efficient Data Processing

Hamza Errahmouni Barkam*
herrahmo@uci.edu
University of California, Irvine
Irvine, California, USA

Tamoghno Das
University of California, Irvine
Irvine, California, USA

Prathyush Poduval
University of California, Irvine
Irvine, California, USA

SungHeon Jeong
University of California, Irvine
Irvine, California, USA

Calvin Yeung
University of California, Irvine
Irvine, California, USA

Mostafa Solitan
University of California, Irvine
Irvine, California, USA

Mohsen Imani*
University of California, Irvine
Irvine, California, USA
m.imani@uci.edu

ABSTRACT

In machine learning (ML), near-sensor AI is transforming edge computing by reducing response times and data transmission, ultimately saving energy and bandwidth. Despite challenges like limited computational resources and the need for transparent decision-making, this approach aims to enhance the intelligence and autonomy of edge devices. Our research presents a novel framework that adds a layer of abstract intelligence to sensors, boosting system efficiency and accuracy through transparent, interpretable sub-symbolic AI. We combine Bayesian algorithms with hyperdimensional computing (HDC), inspired by the human brain's operational efficiency, to deliver an energy-efficient solution matching the accuracy of traditional cloud systems without constant server dependence. This framework uses a binary classifier with Bayesian insights to choose the best data processing location—locally or in the cloud—adapting to data environments. Our method ensures cloud-level performance while significantly reducing energy consumption, improving the sustainability of sensor-based systems. It also enables continual adaptation and learning directly at the sensor level, enriching cloud models with fresh edge insights. Our results have shown to bridge the gap from around 38% quality loss between the standalone near-sensor HDC model and the SOTA cloud-based model to improve the quality loss to only 9% while simultaneously saving 45.34% of energy by not using the cloud. This framework paves the way for more sustainable, efficient, and accurate edge computing in the ML landscape by bridging the gap between simple near-sensor models and their advanced cloud-based counterparts.

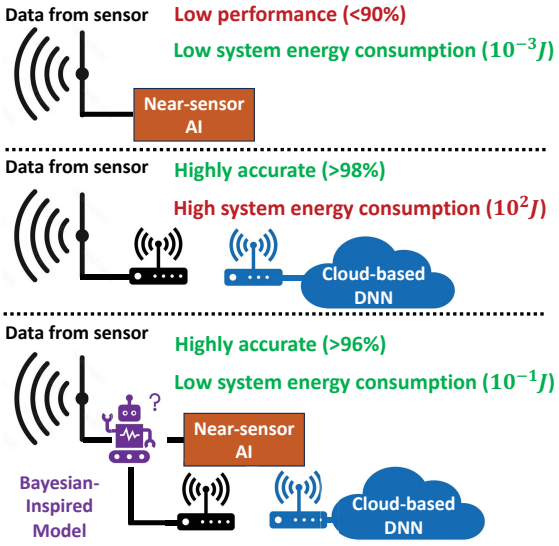

Figure 1: AI Deployment Scenarios in Near-Sensor Environments: This figure compares three models: a basic near-sensor model offering low energy and latency with limited accuracy; an edge server model with high accuracy but increased latency and computational needs; and our framework, which uses a Bayesian-enhanced Hyperdimensional Computing (HDC) model to balance efficiency and accuracy, reducing cloud reliance by only sending low-confidence data to the cloud.

1 INTRODUCTION

As the reach of machine learning (ML) stretches into diverse sectors, embedding AI directly at the point of data collection, known as near-sensor AI, is gaining traction as a critical driver for advanced, sustainable edge computing[1] due to its ability in reducing response times, enhancing immediate data analysis, abstract attention, and significantly lowering the amount of data needing to be sent over networks, thus conserving bandwidth and energy. Such integration is seen as a holistic way to boost the capabilities of edge devices, making them more intelligent and autonomous.

However, this endeavor is not without its challenges[2]. Adapting complex ML models to function efficiently right beside the sensors introduces a range of issues more complex than edge AI, from the struggle with limited processing power and energy availability inherent in many edge devices to potential data loss due to compression techniques like quantization and most notably, concerns about how understandable and transparent the AI's decisions are. Lightweight ML models, also known as tinyML, which typically consist of scaled-down Deep Neural Networks (DNNs) or

ICCAD '24, October 27–31, 2024, New York, NY, USA

ACM ISBN 979-8-4007-1077-3/2024/10.
https://doi.org/10.1145/3676536.3676684

quantized versions of larger models, often suffer from a marked accuracy reduction compared to their full-sized counterparts on cloud or edge servers due to the difficulty in making them compatible with efficient and noisy hardware found near the sensor. Amidst the complexities of integrating AI into edge devices, the appeal of Hyper-Dimensional Computing (HDC) models is gaining traction for its promising role in near-sensor AI applications[3]. Inspired by the human brain's remarkable efficiency and adaptability in processing information, these models provide a blueprint for developing interpretable, energy-efficient AI systems requiring minimal computational power[4–8]. Hyperdimensional computing, which mirrors the brain's capacity to function in vast, high-dimensional spaces, can easily manage intricate patterns[9]. We propose a system that employs an HDC-based near-sensor model designed for real-time data processing, as seen in Figure 1, with whom we address the inherent limitations of these compact IoT devices.

Despite the promise these models hold[10–14], they have encountered challenges in matching the performance levels of state-of-the-art deep neural networks, particularly in tasks that demand detailed data analysis and high accuracy. This shortfall can be remediated by balancing local data inference on the device with HDC models, which is essential for minimizing latency against using more powerful, cloud-based models that are more accurate at the cost of introducing communication overheads, potential delays, and exponentially higher system energy consumption. This discrepancy highlights a significant trade-off: while situating models close to data sources reduces latency and can lower some networking costs, it also necessitates occasional reliance on cloud-based models to achieve higher system accuracy. Deploying machine learning models across diverse computing environments involves optimizing the locations for inference. Recent research has explored using Reinforcement Learning (RL) for resource allocation and task distribution. However, these strategies often come with high computational and energy costs, making them unsuitable for scenarios with limited power and computational resources, particularly near the sensor[15]. This presents a unique challenge for these constrained devices: to design an efficient, low-cost, immediate decision-maker on whether to run inference tasks on the costly yet accurate server or by relying on compact models near the sensor that are not as accurate but efficient.

In response to these inherent challenges, our work introduces a framework designed to close the gap concerning the performance of cloud-based processing without constant reliance on server infrastructure. Recognizing the higher computational demands of Bayesian algorithms compared to conventional Deep Neural Network (DNN) models, our framework strategically employs a binary classifier with Bayesian features, making it a viable solution in scenarios where DNNs near sensors are impractical due to feasibility concerns. This classifier, by extracting and analyzing insights from both near-sensor and cloud models, assesses the reliability of the near-sensor model's predictions to decide whether to process data locally or transmit it to the cloud for a more accurate evaluation. Central to our approach is the innovative application of Bayesian techniques on Hyperdimensional Computing (HDC), marking a significant departure from traditional methods. For the hardware acceleration of the framework, an Analog Compute-in-Memory (CiM) architecture is used, with FeFETs as the devices constituting the memory cells, which have intrinsic conductance variations, which exhibit process-induced stochastic variations [16]. Typically, for DNNs utilizing in-memory computing on such memristive crossbars, these variations cause a loss in inference accuracy. However, using Bayesian inference, which operates on probability distributions instead of exact values, we can leverage the conductance variations as the noise, which makes our classification robust. Using HDC-based Bayesian classifiers further enhances the robustness of decisions due to their holographic properties and intrinsic redundancy. Supported by three advanced algorithms, our framework equips the binary classifier with the necessary training data to navigate ideal (noiseless) and practical (noisy) data streams. This dual capability underscores our framework's adaptability to various data conditions, paving the way for a new foundational mathematical model that redefines efficiency and accuracy in data processing across constrained computing landscapes. In this paper, we concentrate on the healthcare dataset PPG-DaLiA, which exemplifies a scenario where near-sensor devices— specifically, a wearable device with limited computation adjacent to its sensor—allow the Bayesian model and a compact classifier to coexist. Simultaneously, these devices have network capabilities to transmit complex samples to the cloud for selective predictions. Our performance was evaluated using various heuristics to train the Bayesian binary model, which resulted in accuracy improvements of up to 29.08% while reducing power consumption by up to 45.34%.

2 RELATED WORK

2.1 Hyperdimensional Computing

HDC has demonstrated proficiency in addressing diverse tasks and datasets, establishing itself as a robust framework well-suited for applications requiring lightweight, online learning and highly efficient training[17–20]. A noteworthy attribute contributing to HDC's popularity is its resilience against model weights, precisely the dimensions of its hypervectors, as evidenced in prior works[21–23]. This adaptability aligns seamlessly with CiM technologies, known for minimizing energy consumption and reducing delays by mitigating data movement challenges associated with traditional von Neumann architectures[24, 25]. Furthermore, HDC exhibits resilience in handling non-idealities coming from CiM hardware, as corroborated by recent literature[26–30]. In the domain of HDC applications, a notable domain is time series data and multi-classification, where its efficacy has been particularly impressive[31, 32]. However, current complexities, energy consumption, and delays are closely tied to the model's class count and dimensionality, rendering computational costs incompatible with simple data-generating sensors. Proposing a binary classifier within the sensor's architecture, focused on discerning data significance, emerges as a promising alternative to address these challenges.

2.2 Healthcare Activity Monitoring

In healthcare activity monitoring, our attention is directed towards the PPG-DaLiA [33] dataset. PPG-DaLiA, designed for PPG-based

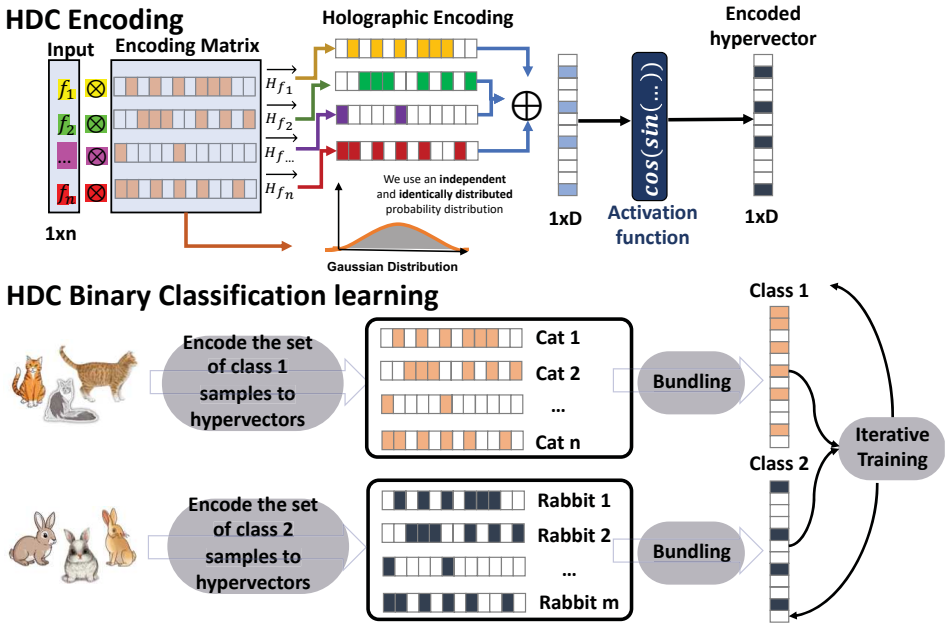

Figure 2: Hyperdimensional Computing (HDC) Workflow Simplified. This figure outlines HDC's key phase: encoding, where input data is transformed into high-dimensional vectors via randomized mapping. Creating robust hyperspace representations and training involving grouping similar data points into class hypervectors, refined through validation to improve accuracy. The workflow highlights HDC's efficiency and simplicity, employing parallelizable mathematical operations for streamlined processing.

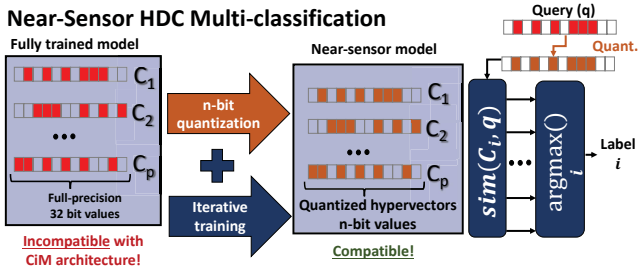

Figure 3: HDC Model Inference for CiM Use. This figure illustrates optimizing a 32-bit precision HDC model for Computing in Memory (CiM) systems. The model is first quantized and then undergoes iterative training to refine class hypervectors, ensuring the retention of crucial information. Finally, query hypervectors are quantized, and their similarity to class hypervectors is evaluated to determine the most likely class label, maintaining accurate performance within CiM's precision limitations.

heart rate estimation, captures data from subjects engaged in various activities under real-life conditions. In state-of-the-art approaches, PPG exhibits optimal performance with a CNN ensemble[33]. We emphasize introducing transparency and interpretability into HDC, which sets it apart from the often black-box-like decision-making processes of deep neural networks. Works such as [34–36] contribute to reasoning in HDC or Genome Sequence Detection and advance the concept of Capacity, offering insights into model performance using hyperparameters before training or deployment. This transparency is crucial, especially in sensitive domains like healthcare, where understanding and interpreting model outputs are essential. By adopting HDC, we address ethical concerns associated with black-box models, ensuring that our framework remains accessible and interpretable for healthcare applications. This approach optimizes resource utilization in sensor-driven scenarios and aligns with the growing need for responsible and interpretable AI solutions in critical domains.

2.3 Bayesian Inference on resource-constrained environments

Bayesian inference techniques can equip models with a means for uncertainty estimation, which can be crucial for decision-making in resource-constrained environments, especially in Safety-critical sensory applications, such as medical diagnosis, which require precise decisions from limited and noisy data. It adds to the models' adaptability in dynamic environments like near-sensor computations and robustness by providing uncertainty and incorporating new evidence to update beliefs over time, even if the data is limited. Bayesian neural networks excel by providing accurate predictions along with uncertainty assessment[37]. Despite their superior accuracy, deep neural networks consume significant power, making them less suitable for edge devices with limited resources[38]. To improve DNNs on edge, the proposed ways face the problem of customization of the networks to fit each purpose and different devices, which requires professional knowledge and a lot of experiments. Other approaches use a Bayesian search at the end to minimize the number of comparisons needed. Another proposed approach uses support vector machines for their effectiveness in

high-dimensional spaces and their ability to handle non-linear decision boundaries[39]. Equipping them with Bayesian inference offers robustness to handle noisy or uncertain data, enhanced model interpretability, and the ability to estimate prediction uncertainties[40, 41]. Recent advancements in brain-inspired computing have paved the way for highly efficient Machine Learning (ML) algorithms rather than Deep neural networks or SVMs. HyperDimensional Computing (HDC) research has led to significant speedups in ML model training and inference compared to deep learning methods. However, existing HDC-based algorithms often lack uncertainty estimation. However, some models like DiceHD enable uncertainty estimation on the HDC regression models while being much faster and using less power than Bayesian neural networks (BNNs) [42]. This can have many applications as they provide efficient dataset annotation and lower labor costs, unlike other classifiers that need gradient or probabilistic so that they can be easily integrated into any near sensor computing architecture [43].

2.4 Compute-in-memory-based acceleration of Bayesian classifiers:

Initially, Bayesian classifiers were computationally intensive due to the need for repeated probability calculations, making them unsuitable for real-time near-sensor applications. The pursuit of Compute-in-Memory (CiM) technologies to expedite Bayesian inference has been explored in the literature using multiple approaches, with Gao et al. [44] addressing the resilience of memristor crossbar arrays for Bayesian inference, introducing techniques to improve computational reliability. The work presented in [45] explores a memristor-based Bayesian machine, demonstrating the potential speed and energy advantages of using CiM hardware. However, these designs depended on an LFSR-based random-number generation block, which incurred extra peripheral circuitry.

Further advancements are seen in Shukla et al. [46], who propose a CiM architecture with Monte Carlo dropouts for edge-based Bayesian inference, ingeniously integrating stochasticity with hardware efficiency. Despite this, the energy costs of Monte Carlo methods are prohibitively high for near-sensor integration. Manna et al. [16] perform extensive device characterization of FeFETs to leverage the intrinsic noise due to conductance variations for probabilistic deep learning using in-memory computing. However, all methods above employ BNNs, which are resource-intensive due to the need for multiple layers of computation. DiceHD [42] presents the first exploration of integrating the robustness of Bayesian inference with the hardware efficiency of HDC for uncertainty estimation in HDC-based regression algorithms. However, it is primarily focused on regression tasks instead of learning tasks. It is inefficient for hardware implementation as it requires twice the memory overhead to save the prior distribution and the hypervectors separately. Thus, the need for a more efficient method of performing classification tasks near sensors with minimal area and energy consumption in resource-constrained environments is imminent.

3 FRAMEWORK

3.0.1 Hyperdimensional Learning. We utilize hyperdimensional learning for direct encoded data processing. In our framework, HDC identifies patterns in training data by grouping them according to their labels. Instead of simply combining all encoded data, our approach involves incrementally adding each data point to the class hypervectors based on the novelty of the information it introduces. To prevent hypervector saturation, we limit or avoid adding data to class hypervectors if a data point is already present. Furthermore, no updates occur to prevent overfitting when predictions match expected outputs. This adaptive update strategy gives greater weight to uncommon patterns, refining the final model. Importantly, this technique eliminates the need for time-consuming iterative training. Let us assume $\vec{\mathcal{V}}$ as a new training data point. The model will compute the cosine similarity of $\vec{\mathcal{V}}$ with both class hypervectors, in this case, datapoint of interest ($\vec{C}_1$) or not $\vec{C}_0$. We compute similarity of a data point with $\vec{C}_i$ as: $\delta(\vec{\mathcal{V}}, \vec{C}_i)$. Instead of naively adding a data point to the model, HDC updates the model based on the δ similarity. If the model incorrectly returns the label 1 for an encoded query $\vec{\mathcal{H}}$, the model updates as follows:

$$\vec{C}_0 \leftarrow \vec{C}_0 + \gamma\,(1 - \delta_0) \times \vec{\mathcal{V}} \tag{1}$$

$$\vec{C}_1 \leftarrow \vec{C}_1 - \eta\,(\delta_1) \times \vec{\mathcal{H}} \tag{2}$$

where γ and η are a learning rates. A large δ_l indicates that the input is a common data point already existing in the model. Therefore, our update adds a small portion of the encoded query to the model to eliminate model saturation ($1 - \delta_l \simeq 0$). In inference, HDC checks the similarity of each encoded test data with the class hypervector in two steps. The first step encodes the input (the same encoding used for training) to produce a query hypervector $\vec{\mathcal{V}}$. Then, we compute the similarity (δ) of $\vec{\mathcal{V}}$ and all class hypervectors. Query data gets the label of the class with the highest similarity.

3.0.2 Hyperdimensional Uncertainty Estimation. The HDC encoder generates a random large D–dimensional representation $\vec{\mathcal{H}}_f$ of a d–dimensional feature vector $\vec{f}$, such that the similarity of the hypervectors approximates a kernel $k(\vec{x})$

$$\delta(\vec{\mathcal{H}}_f, \vec{\mathcal{H}}_g) = k(\vec{f} - \vec{g}). \tag{3}$$

The encoding is generated by sampling a random matrix $M_{D\times d}$ from a probability distribution $p(\vec{\omega})$, and then defining $\vec{\mathcal{H}}_f = \exp(iM\vec{f})$. If $p(\vec{\omega})$ is chosen to be the Fourier transform of the kernel $p(\vec{\omega}) = \int k(\vec{x}) e^{i\vec{\omega}\cdot\vec{x}} d\vec{x}$, then Bochner's theorem guarantees that the similarity relation of Eq. 3 holds. Then, the corresponding class vectors are constructed using Eq. 2.

However, one key problem is choosing the kernel $k(\vec{x})$ appropriately. Different kernel choices can have small variations in the model's accuracy, and as a result, there will be some inherent uncertainty in the predictions. To estimate the uncertainty, we place a prior distribution over a subspace of possible kernels spanned by a set of basis kernels. We do this by first choosing N basis kernels labeled $k_i(\vec{x})$, with corresponding probability distributions $p_i(\vec{\omega})$. We will choose each p_i to be a one-dimensional distribution, identical over each component. We then construct a random kernel, characterized by $\vec{w} = (w_1, w_2, ..., w_N)$ (with $\sum_i w_i = 1$ and $w_i = 0$), defined as $k_w = \sum_i w_i k_i$, and the corresponding probability distribution is given by $p_w = \sum_i w_i p_i$. We define a prior on $\vec{w}$ through a two-stage

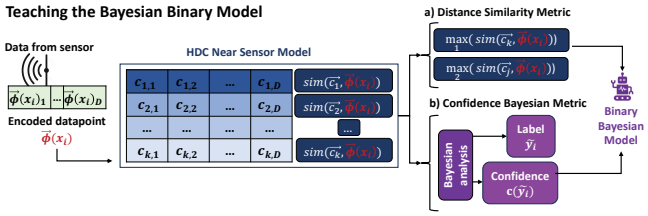

Figure 4: Overview of our adaptive framework integrating Bayesian hyperdimensional computing (HDC). An ensemble of Bayesian binary classifiers decides whether data should be classified locally or sent to the cloud. The HDC multi-classifier provides initial predictions, and a Bayesian regression model evaluates the confidence range between class hypervectors, delegating complex cases to the cloud.

process of first sampling w_i from a uniform Uniform(0, 1) distribution and then normalizing the values to $w_i \leftarrow w_i / \sum_i w_i$. Finally, for the uncertainty estimation, we construct multiple random models by sampling different values from the prior of $\vec{w}$ and characterizing the distribution of the resulting accuracies of the model. The sampling of p_w can be performed efficiently by first calculating the corresponding cumulative function $c_w(z) = \int_{-\infty}^{z} dx p_w(x)$. Then, to sample from p_w, we can instead sample from $c_w^{-1}(U)$, where U is a Uniform(0, 1) distribution. To characterize the uncertainty of the model, we perform a simple Bayesian analysis of the results over the multiple models. Given the training data $\mathcal{D}$, and an input $\vec{x}$, the probability of a certain class is given by $P(Y|\vec{x}, \mathcal{D})$, which can be written using Bayes Theorem as

$$P(Y|\vec{x}, \mathcal{D}) = P(Y|\vec{x}, \mathcal{D}, \vec{w})P(\vec{w}). \tag{4}$$

For a given set of parameters $\vec{w}$, the probability of predicting a certain class in the model is simply the indicated function of the class c with maximum similarity $s(c)$, and so we have

$$P(Y = i|\vec{x}, \mathcal{D}, \vec{w}) = \delta\left(i = \text{argmax}_c s(c)\right), \tag{5}$$

where the δ–function is 0 if the condition is false, and 1 otherwise. Thus, the probability of a class prediction is given by

$$P(Y|\vec{x}, \mathcal{D}) = \delta\left(i = \text{argmax}_c s(c)\right) P(\vec{w}). \tag{6}$$

In our work, we estimate the distribution $P(Y|\vec{x}, \mathcal{D})$ by sampling multiple models through samples of $\vec{w}$ and then numerically calculating the fraction of times $Y = i$ was predicted. This will give us a measure of confidence in our prediction through probabilistic analysis.

3.1 Framework

Our framework employs a novel approach that utilizes hyperdimensional computing (HDC) for efficient binary classification, specifically designed for scenarios with limited data. At its core, this framework integrates a Bayesian HDC binary classifier to make a strategic decision: whether to classify data locally at the sensor or send it to the cloud for more refined processing, as shown in Figure 4. By emphasizing transparency and mathematical rigor, we leverage Bayesian principles to improve upon the near-sensor setup.

HDC Binary Classification Model: For the first configuration, we use the HDC multi-classifier to generate data predictions, which are then compared to the cloud model's outputs to establish a ground truth baseline. This comparison reveals discrepancies and helps identify situations where the near-sensor model may struggle, requiring the cloud model's intervention. Our approach enriches the training of the HDC binary classifier by drawing on three key sources of information. This step is crucial, as the HDC model's strength in few-shot learning makes it an excellent option for handling unbalanced datasets, which are typical in our target applications. These datasets often contain sparse instances where the near-sensor model misclassifies.

HDC Bayesian Binary Classification Model: Instead of relying on a standard binary HDC classifier, we employ an ensemble of Bayesian binary classifiers to make predictions, greatly enhancing reliability and adaptability. The reliability and adaptability of the Bayesian HDC ensemble are crucial for healthcare near-sensor AI devices, where accurate interpretation of physiological signals is vital. Patients' conditions can change rapidly, so a dependable classification system is essential to avoid misinterpretation and inappropriate medical actions. The system must also adapt to individual differences in patient data, recognizing unique patterns and making accurate predictions across varying conditions. With limited computational resources, near-sensor devices must efficiently classify simple cases locally while sending complex signals to the cloud for advanced analysis, conserving power and reducing delays. This ensures swift, accurate decision-making, which is essential for patient safety and timely intervention.

Distance-based Bayesian Model: In our approach, we implement a Bayesian regression model using HDC to predict, with a specified confidence range, the distance between the two class similarity hypervectors closest to the query. One of these represents the correct label, while the other indicates an incorrect classification. This analysis helps determine how near the query hypervector is to each class, allowing us to use Bayesian confidence to evaluate classification risk. A predefined threshold decides whether the prediction should be made at the sensor or escalated to the cloud, as depicted by Figure 5. This method ensures that local predictions are reliable, while more complex or uncertain cases are directed to the cloud for detailed processing. HDC's mathematical principles and transparency are vital in this framework, making it easy to spot ambiguous situations where a sample is too close to the neighboring hypervectors. This clear decision-making process is crucial in healthcare, where understanding the reasoning behind predictions builds trust and ensures quick, accurate responses. By leveraging the transparency and analytical strengths of this model, our Bayesian regression approach ensures accurate classifications, reducing computational load near the sensor while prioritizing patient safety.

In general, our adaptive framework harnesses the strengths of HDC's mathematical foundation and Bayesian modeling to achieve improved accuracy, transparency, and efficient data classification. The seamless integration of near-sensor classification and cloud processing enhances system performance while maintaining low power consumption and high adaptability.

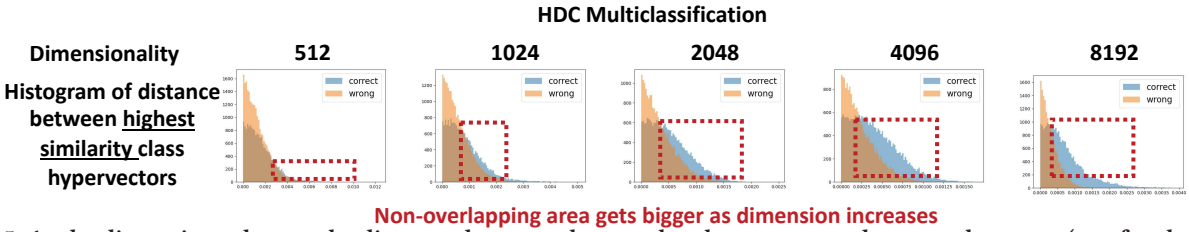

Figure 5: As the dimensions change, the distances between the two class hypervectors closest to the query (one for the correct label and one for the incorrect label) follow a predictable pattern. The Bayesian regression model measures this distance, using a preset threshold to assess whether it's too risky to classify near the sensor. If the threshold is exceeded, the prediction is sent to the cloud for further analysis, ensuring accurate and reliable classification.

4 HARDWARE ARCHITECTURE

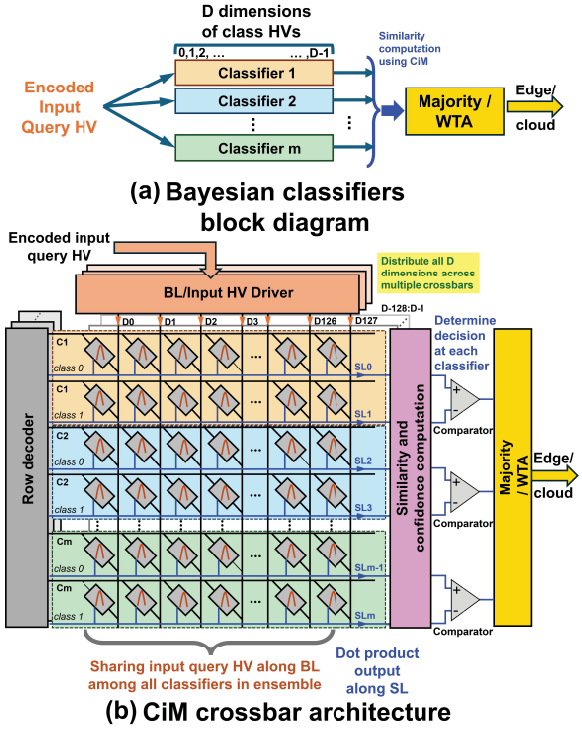

Figure 6: a) Block diagram of Bayesian HDC binary classifiers, and b) CiM architecture showing the corresponding mapping onto FeFET crossbars.

To tackle the shortcomings of existing works trying to perform Bayesian inference near-sensor, this section describes the novel hardware architecture used to implement the ensemble of Bayesian HDC binary classifiers, which perform intelligent sensing and dynamically determine the mode of operation between an HDC-based classification model at the edge and cloud-based processing (Fig. 6(a)). Our method leverages the majority voting mechanism among the classifiers, but could be equivalently replaced by a Winner-Takes-All (WTA) circuit. Each classifier consists of long hypervectors (HVs) that necessitate extensive memory for storage. Typically, these HVs require a large number of columns in memory crossbars while only two rows are necessary due to the final output being a decision between two classes. However, mapping each classifier on separate memory crossbars results in suboptimal memory utilization, throughput, and energy efficiency.

To address these challenges, we propose a novel scheme for mapping these Bayesian HDC binary classifiers onto memory crossbars to utilize the improvements in density, parallelism, and energy efficiency of Computing in Memory (CiM). Given the constraints on physical dimensions in memory crossbar fabrication, our strategy involves distributing the dimensions of the HVs along the columns of multiple crossbars as shown in Fig. 6 (b). The holographic nature of HDC allows for independent computation along each dimension, thus enabling a seamless distribution without impacting the accuracy of computations. This is in contrast to neural networks, where inter-layer dependencies necessitate sequential data processing and prevent such independent distribution of weights. By mapping the weights corresponding to each classifier onto consecutive rows within the same crossbar, we significantly improve memory utilization, throughput and energy efficiency. For Bayesian inference, the weights include additional noise due to sampling from a probability distribution, which lends robustness and the capability to retain accuracy despite uncertainty. In the proposed architecture, instead of having a separate random number generator, we utilize the stochastic conductance variations in FeFET devices characterized extensively in [16] as the additional noise on the weights. This approach is more efficient in terms of energy and area, hence making the framework well-suited for near-sensor integration.

5 EVALUATION

5.1 Experimental Setup

Our integrated system, which combines a Bayesian approach, an edge computing model, and cloud analytics, has been effectively deployed using the PyTorch framework. To gauge the effectiveness of our approach, we undertook a detailed evaluation using a renowned dataset for healthcare wearable device applications, such as PPG-Dalia. The assessment focused on key performance indicators relevant to machine learning, including accuracy and overall energy usage. The latter considers the sensor's energy demands, the Bayesian analysis's computational cost, and the energy implications of choosing between local processing and cloud-based computation, including data transmission costs to the edge. The first benchmark of our analysis centers on a standalone Hyperdimensional Computing (HDC) model strategically positioned near the sensor. This model operates with an 8192 dimensionality, applying a Gaussian distribution for its randomized encoding. Designed with sensor

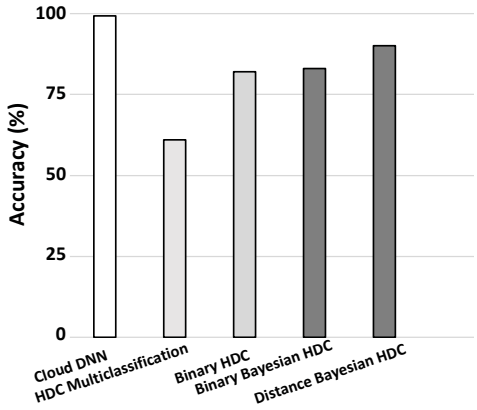

Figure 7: Comprehensive summary of system accuracy for various HDC configurations at their best-performing dimension. It highlights the differing performance levels across multiple configurations, demonstrating how each model's design influences its effectiveness in handling system total accuracy.

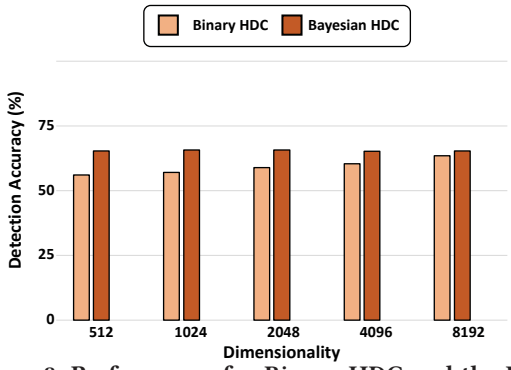

Figure 8: Performance for Binary HDC and the Bayesian HDC Models across different dimensions.

proximity in mind, the model is fine-tuned using a memory-centric computing architecture alongside advanced quantization and our novel learning methods to reduce information loss, ensuring optimal performance for edge deployments. In contrast, the second benchmark of our analysis delves into a more conventional setup, where data is processed remotely on an edge server equipped with a complex, multi-layered Deep Neural Network (DNN) architecture. This system is intricately designed to efficiently manage and interpret high-dimensional data, employing a strategic combination of ReLU activation functions, a 0.3 dropout rate, and batch normalization to enhance accuracy and throughput despite the inevitable increase in computational demand and response time. Through this comparative analysis, our goal is to highlight the flexibility and effectiveness of our proposed solution, demonstrating its ability to merge the immediacy and energy efficiency of edge computing with the analytical depth and accuracy of cloud-based processing.

5.2 System Accuracy

Our study investigates the Bayesian Hyperdimensional Computing (HDC) model's capability to predict whether the data points will be correctly predicted by the HDC multiclassification model or not. A critical aspect of this examination is determining how effectively the model distinguishes between classes, particularly when data points are similarly close to multiple class hypervectors, which could present classification challenges which is dependent on the dimensionality. We conducted experiments across various dimensions—512, 1024, 2048, 4096, and 8192—to assess how these dimensionalities impact the model's accuracy. Figure 7 shows a comprehensive summary of system accuracy across different configurations and for the best-performing dimension. Notably, the Distance-based Bayesian HDC model shows exceptional accuracy, particularly at higher dimensions, peaking at 91.41% at 4096 dimensions and maintaining a high rate at 8192 dimensions. This model's design to analyze distance histograms allows it to efficiently handle scenarios where data points are close to multiple hypervectors, enhancing its discrimination capabilities.

The findings underscore the significant impact of utilizing HDC classifiers near the sensor. By implementing these models, the system not only maximizes local processing capabilities but also ensures that data requiring more detailed analysis is accurately identified and sent to the cloud. This strategic balance optimizes both computational efficiency and accuracy, which is critical for real-time applications such as in healthcare monitoring, where timely and precise data classification can be crucial. Each configuration was tested 10 times to ensure the robustness of our findings, with the average accuracy reported to account for variability, confirming the reliability of the observed trends.

We analyzed near-sensor model usage across various configurations, revealing distinct patterns of local versus cloud processing dependency. The Bayesian HDC and Standard Binary HDC models show moderate near-sensor reliance, with about 60% of processing managed locally, blending efficiency with cloud capabilities for complex decisions. The Distance-based Bayesian HDC model displays fluctuating near-sensor usage from 51.97% at 512 dimensions to 32.75% at 4096 dimensions, indicating dynamic processing allocation based on data complexity and the model's capability within sensor constraints. Near-sensor and cloud processing trends significantly impact system accuracy. Models with high near-sensor reliance, like the Multiclassification HDC, bypass latency and bandwidth issues but may sacrifice analytical depth. Conversely, models like the Distance Histogram Bayesian HDC, which selectively use cloud resources, handle more complex challenges with their enhanced power, showing variable accuracy. This balance between local and cloud processing is crucial, particularly in healthcare, where real-time data processing is vital for patient monitoring and decision-making.

5.3 Dimensionality exploration for near-sensor binary model

This subsection examines the Bayesian model's hyperparameters directly within the near-sensor environment, aiming to gather data for the binary classifier to decide whether data points should be processed locally or escalated to the cloud. Our primary goal is to assess the model's ability to produce meaningful confidence and accuracy metrics, which are essential inputs for the binary classifier. The classifier's role is crucial as it gauges the confidence level of each

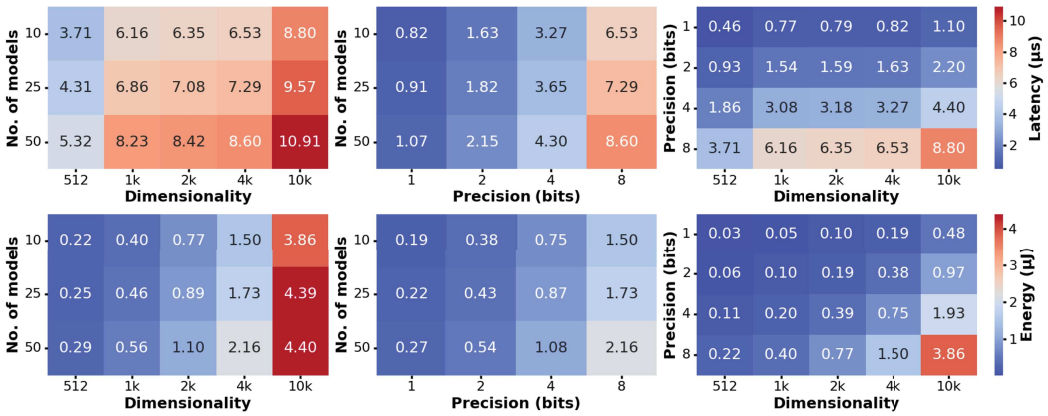

Figure 9: Latency and energy distribution with no. of models, dimensionality, and bit-precision

data point and employs a preset threshold to determine whether to process the data locally or offload it to the cloud for further analysis. The evaluation was conducted across various dimensions (512, 1024, 2048, 4096, and 8192) in the Bayesian classifier, and can be found in Figure 8. This comprehensive analysis reveals how different dimensional settings affect the classifier's performance, helping identify the most efficient configuration for processing data. The evaluation examined the model's detection capabilities across five different dimensions: 512, 1024, 2048, 4096, and 8192. In the standard binary HDC model, performance gradually improved as the dimensionality increased, starting at 56.09% accuracy at 512 dimensions and reaching 63.47% at 8192. However, the base Bayesian HDC model consistently outperformed the standard binary model across all dimensional settings. It maintained a relatively stable accuracy rate, starting at 65.36% for 512 dimensions and varying only slightly to 65.33% at 8192 dimensions. This demonstrates the Bayesian model's robustness and reliability, which is crucial in ensuring data points are classified correctly and only escalated to the cloud when necessary. The stable performance across dimensions showcases the adaptability of Bayesian HDC, allowing it to make more informed and accurate classification decisions even with varying amounts of data.

5.4 Energy and latency comparison

For the hardware evaluation of the proposed framework, Neurosim [47], a tool for modeling analog Compute-in-Memory (CiM) hardware, is employed to simulate bitwise-dot product operations. These operations are integral to performing cosine similarity calculations in the associative search phase of Hyperdimensional Computing (HDC) used for inference. To account for the Bayesian inference overhead, the experimental conductance variation characterization for different device sizes and programming voltages on industry-scale FeFETs is taken from [16], and the device-level variations are incorporated into the training process for the Bayesian classifiers. Thus, we do not need a separate random number generator to generate the probability distribution and sample a value each cycle. The evaluation systematically measures latency and energy consumption across varying model configurations—specifically concerning bit precision and dimensionality of hypervectors, and number of models used for the ensemble.

From the results in Fig. 9 it is seen that increase in dimensionality significantly escalates both latency and energy consumption, evidenced by a jump in latency from 3.71μs to 8.80μs and energy consumption from 0.22μJ to 3.86μJ when dimensionality scales from 512 to 10k with ten models. Similarly, an increased number of models raises both latency and energy consumption across all dimensionalities, albeit in a sub-linear fashion. This trend suggests that the computational costs associated with more complex model ensembles increase at a more gradual rate.

Fig. 9 also illustrates the substantial impact that precision has on both latency and energy consumption. Specifically, as precision increases from 1 bit to 8 bits, there is a discernible escalation in both metrics. For instance, with ten models at a dimensionality of 512, latency climbs from 0.46μs to 1.10μs, and energy consumption rises from 0.03μJ to 0.48μJ. This trend is consistent across all model configurations, suggesting that higher precision levels, while potentially improving computational accuracy, do impose a certain overhead on system resources. However, the latency and energy consumption for all cases are quite tiny, proving that our framework is highly suitable for near-sensor integration. This evaluation emphasizes the critical tradeoff between model accuracy and compute efficiency, highlighting the need for careful setting of hyperparameters for model configuration in resource-constrained near-sensor environments with limited power and area budget, while still meeting the latency requirements for real-time data processing.

6 CONCLUSIONS

In conclusion, our research introduces a novel framework that incorporates a layer of abstract intelligence into sensors using hyperdimensional computing (HDC). This allows near-sensor AI deployment to overcome critical challenges by enhancing autonomy, efficiency, and responsiveness. Our framework, which utilizes transparent and interpretable sub-symbolic AI, optimizes data collection and processing directly at the source, resulting in an efficient system structure. This significantly improves accuracy while reducing the need for extensive data transmission, conserving bandwidth, and minimizing latency. These improvements are crucial in achieving a more sustainable, efficient, and accurate system, making this framework an essential development in the rapidly advancing field of near-sensor computing.

ACKNOWLEDGMENTS

This work was supported in part by the DARPA Young Faculty Award, the National Science Foundation (NSF) under Grants #2127780, #2319198, #2321840, #2312517, and #2235472, the Semiconductor Research Corporation (SRC), the Office of Naval Research through the Young Investigator Program Award, and Grants #N00014-21-1-2225 and #N00014-22-1-2067. Additionally, support was provided by the Air Force Office of Scientific Research under Award #FA9550-22-1-0253, along with generous gifts from Xilinx and Cisco.

REFERENCES

[1] F. Zhou and Y. Chai, "Near-sensor and in-sensor computing," Nature Electronics, vol. 3, no. 11, pp. 664–671, 2020.

[2] G. Plastiras et al., "Edge intelligence: Challenges and opportunities of near-sensor machine learning applications," in *2018 IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2018, pp. 1–7.

[3] S. Yun et al., *Hypersense: Accelerating hyper-dimensional computing for intelligent sensor data processing*, 2024. arXiv: 2401.10267 [cs.AR].

[4] A. Hernandez-Cane et al., "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *DATE*, IEEE, 2021, pp. 56–61.

[5] P. P. Poduval et al., "Hdqmf: Holographic feature decomposition using quantum algorithms," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 978–10 987.

[6] Y. Ni et al., "Brain-inspired trustworthy hyperdimensional computing with efficient uncertainty quantification," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 01–09.

[7] M. Imani et al., "Exploring hyperdimensional associative memory," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, IEEE, 2017, pp. 445–456.

[8] Z. Zou et al., "Biohd: An efficient genome sequence search platform using hyperdimensional memorization," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, 2022, pp. 656–669.

[9] P. Kanerva, "Binary spatter-coding of ordered k-tuples," in *ICANN*, 1996.

[10] P. R. Genssler et al., "Modeling and predicting transistor aging under workload dependency using machine learning," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 70, no. 9, pp. 3699–3711, 2023.

[11] H. Amrouch et al., "Brain-inspired hyperdimensional computing for ultra-efficient edge ai," in *2022 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2022, pp. 25–34.

[12] H. E. Barkam et al., "Invited paper: Hyperdimensional computing for resilient edge learning," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–8.

[13] Z. Zou et al., "Eventhd: Robust and efficient hyperdimensional learning with neuromorphic sensor," Frontiers in Neuroscience, vol. 16, p. 858 329, 2022.

[14] H. Chen et al., "Scalable and interpretable brain-inspired hyper-dimensional computing intelligence with hardware-software co-design," in *2024 IEEE Custom Integrated Circuits Conference (CICC)*, 2024, pp. 1–8.

[15] G. M. Skaltsis et al., "A survey of task allocation techniques in mas," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021, pp. 488–497.

[16] B. Manna et al., "Variation-resilient fefet-based in-memory computing leveraging probabilistic deep learning," IEEE Transactions on Electron Devices, 2024.

[17] 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.

[18] Z. Zou et al., "Scalable edge-based hyperdimensional learning system with brain-like neural adaptation," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–15.

[19] F. J. Piran et al., *Explainable hyperdimensional computing for balancing privacy and transparency in additive manufacturing monitoring*, 2024. arXiv: 2407.07066 [cs.LG].

[20] H. Chen et al., "Hypergraf: Hyperdimensional graph-based reasoning acceleration on fpga," in *2023 33rd International Conference on Field-Programmable Logic and Applications (FPL)*, 2023, pp. 34–41.

[21] S. Zhang et al., "Assessing robustness of hyperdimensional computing against errors in associative memory : (invited paper)," in *2021 IEEE 32nd International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2021, pp. 211–217.

[22] S. Yun et al., "Hyperdimensional computing for robust and efficient unsupervised learning," in *2023 57th Asilomar Conference on Signals, Systems, and Computers*, 2023, pp. 281–288.

[23] M. Imani et al., "Voicehd: Hyperdimensional computing for efficient speech recognition," in *2017 IEEE international conference on rebooting computing (ICRC)*, IEEE, 2017, pp. 1–8.

[24] M. Imani et al., "Dual: Acceleration of clustering algorithms using digital-based processing in-memory," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, IEEE, 2020, pp. 356–371.

[25] M. Imani et al., "Revisiting hyperdimensional learning for fpga and low-power architectures," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, IEEE, 2021, pp. 221–234.

[26] G. Karunaratne et al., "In-memory hyperdimensional computing," Nature Electronics, vol. 3, no. 6, pp. 327–337, 2020.

[27] A. Kazemi et al., "Achieving software-equivalent accuracy for hyperdimensional computing with ferroelectric-based in-memory computing," Scientific reports, vol. 12, no. 1, p. 19 201, 2022.

[28] H. E. Barkam et al., "Reliable hyperdimensional reasoning on unreliable emerging technologies," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–9.

[29] Q. Huang et al., "A fefet-based time-domain associative memory for multi-bit similarity computation," in *2024 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2024, pp. 1–6.

[30] H. E. Barkam et al., "In-memory acceleration of hyperdimensional genome matching on unreliable emerging technologies," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 71, no. 4, pp. 1794–1807, 2024.

[31] Y. Ni et al., "Neurally-inspired hyperdimensional classification for efficient and robust biosignal processing," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, ser. ICCAD '22, San Diego, California: Association for Computing Machinery, 2022.

[32] S. Aygun et al., "Learning from hypervectors: A survey on hypervector encoding," arXiv preprint arXiv:2308.00685, 2023.

[33] A. Reiss et al., "Deep PPG: Large-Scale heart rate estimation with convolutional neural networks," en, Sensors (Basel), vol. 19, no. 14, 2019.

[34] P. Poduval et al., "Graphd: Graph-based hyperdimensional memorization for brain-like cognitive learning," Frontiers in Neuroscience, p. 5, 2022.

[35] H. E. Barkam et al., "Hdgim: Hyperdimensional genome sequence matching on unreliable highly scaled fefet," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–6.

[36] H. Chen et al., *Hdreason: Algorithm-hardware codesign for hyperdimensional knowledge graph reasoning*, 2024. arXiv: 2403.05763 [cs.AR].

[37] J. P. Bharadiya, "A review of bayesian machine learning principles, methods, and applications," International Journal of Innovative Science and Research Technology, vol. 8, no. 5, pp. 2033–2038, 2023.

[38] D. Bonnet et al., "Bringing uncertainty quantification to the extreme-edge with memristor-based bayesian neural networks," Nature Communications, vol. 14, no. 1, p. 7530, 2023.

[39] Z. Yang et al., "Efficient resource-aware convolutional neural architecture search for edge computing with pareto-bayesian optimization," Sensors, vol. 21, no. 2, 2021.

[40] V. A. Sotiris et al., "Anomaly detection through a bayesian support vector machine," IEEE Transactions on Reliability, vol. 59, no. 2, pp. 277–286, 2010.

[41] P. Sollich, "Bayesian methods for support vector machines: Evidence and predictive class probabilities," Machine learning, vol. 46, pp. 21–52, 2002.

[42] Y. Ni et al., "Brain-inspired trustworthy hyperdimensional computing with efficient uncertainty quantification," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, IEEE, 2023, pp. 01–09.

[43] Y. Ni et al., "Heal: Brain-inspired hyperdimensional efficient active learning," arXiv preprint arXiv:2402.11223, 2024.

[44] D. Gao et al., "Bayesian Inference Based Robust Computing on Memristor Crossbar," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 121–126.

[45] K.-E. Harabi et al., "A memristor-based bayesian machine," Nature Electronics, vol. 6, no. 1, pp. 52–63, 2023.

[46] P. Shukla et al., "MC-CIM: Compute-in-Memory With Monte-Carlo Dropouts for Bayesian Edge Intelligence," en, IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 70, no. 2, pp. 884–896, 2023.

[47] X. Peng et al., "Dnn+ neurosim: An end-to-end benchmarking framework for compute-in-memory accelerators with versatile device technologies," in *2019 IEEE international electron devices meeting (IEDM)*, IEEE, 2019, pp. 32–5.