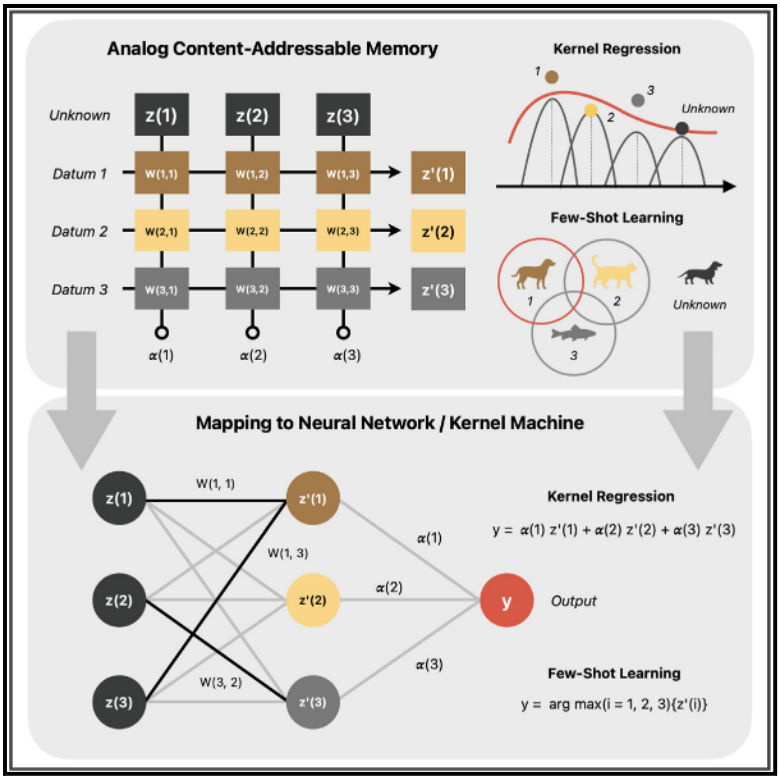
Device

Article

Analog content-addressable memory from complementary FeFETs

Graphical abstract



Highlights

- Proposed ACAM performs parallel search in analog domain with over 40 match windows
- Each cell built on 2 complementary FeFETs, projected to be 3× denser than TCAM
- Improves upon accuracy benchmark for few-shot learning on Omniglot task by 5%
- One-step inference on kernel regression yields 1,000× faster inference than CPU/GPU

Authors

Xiwen Liu, Keshava Katti, Yunfei He, ..., Santosh Kurinec, Pratik Chaudhari, Deep Jariwala

Correspondence

pratikac@seas.upenn.edu (P.C.), dmj@seas.upenn.edu (D.J.)

In brief

The growing trend of overparameterizing deep neural networks shows no sign of slowing down and is beginning to outstrip the capabilities of digital computing, with large language models taking up to 6 months to train on more than 25,000 graphics processing units (GPUs). Parallel search architectures with compute-in-memory (CIM) represent a promising response to this trend by performing crucial similarity comparisons up to 1,000× faster than GPUs, allowing learning on greater amounts of data at a faster rate.

Explore
Early prototypes with exciting performance and new methodology

Liu et al., 2024, Device 2, 100218
February 16, 2024
https://doi.org/10.1016/j.device.2023.100218

CellPress

Device

CellPress
OPEN ACCESS

Article

Analog content-addressable memory from complementary FeFETs

Xiwen Liu,[1,4,5] Keshava Katti,[1,5] Yunfei He,[1] Paul Jacob,[2] Claudia Richter,[3] Uwe Schroeder,[3] Santosh Kurinec,[2] Pratik Chaudhari,[1,*] and Deep Jariwala[1,6,*]

[1]Electrical & Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA
[2]Electrical & Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY, USA
[3]Namlab/TU Dresden, Noethnitzer Strasse 64a, 01187 Dresden, Germany
[4]Thrust of Microelectronics, The Hong Kong University of Science and Technology (Guangzhou), Guangdong, China
[5]These authors contributed equally
[6]Lead contact
*Correspondence: pratikac@seas.upenn.edu (P.C.), dmj@seas.upenn.edu (D.J.)
https://doi.org/10.1016/j.device.2023.100218

THE BIGGER PICTURE Compute-in-memory (CIM) integrates memory and processing units into the same physical location, reducing the time and energy overhead of computing systems to address the increasing demands of artificial intelligence (AI). In particular, AI applications of CIM beyond matrix multiplication, such as parallel search, remain relatively unexplored. Analog content-addressable memory (ACAM) rapidly executes high-accuracy similarity comparisons between inputs and a large number of stored data entries, an operation that lies at the core of large language models, like the Generative Pre-trained Transformer 4 (GPT-4) and Contrastive Language-Image Pre-training (CLIP). With the number of parameters within these models reaching the order of trillions, building efficient in-memory parallel search is more important than ever, offering to open avenues for more comprehensive data processing and richer AI applications ranging from self-attention in transformers to classifier layers for few-shot learning.

SUMMARY

Despite recent advancements in non-volatile memory (NVM) for matrix multiplication, other critical data-intensive operations like parallel search remain largely overlooked. Current parallel search architectures, namely content-addressable memory (CAM), often use binary, which restricts density and functionality. We present an analog CAM (ACAM) cell, built on two complementary ferroelectric field-effect transistors (FeFETs), that performs parallel search in the analog domain with over 40 distinct match windows. ACAM not only offers a projected 3× denser memory architecture than ternary CAM (TCAM) but also yields a 5% increase in inference accuracy on similarity search for few-shot learning simulated with the Omniglot dataset, with an estimated speedup per similarity search of more than 100× when compared to a central processing unit (CPU) and graphics processing unit (GPU) on scaled silicon-based complementary metal-oxide semiconductor (CMOS) nodes. Furthermore, we demonstrate one-step inference on a kernel regression model in ACAM, with simulation results indicating 1,000× faster inference than a CPU and GPU.

INTRODUCTION

In the rapidly evolving field of artificial intelligence (AI), processing vast amounts of data efficiently is critical. Computing architectures that incorporate in-memory processing are gaining traction, as they offer to potentially reduce power consumption and cost.[1] In particular, compute-in-memory (CIM) aims to reduce the energy-intensive data movement limiting traditional computing architectures by integrating memory and processing within the same physical location.[2,3] Recent studies have shown progress in employing non-volatile memory (NVM) devices to accelerate matrix multiplication within memory arrays, in turn aiding execution of complex AI tasks.[4–7] However, expanding these CIM architectures to accommodate other operations intrinsic to AI, notably parallel search, presents a considerable challenge.[6–8] We define parallel search to be a single-step operation of comparing a query vector to an array of *m* stored vectors and returning a degree of match between the query and each stored vector.

Content-addressable memory (CAM) is well-suited to execute operations that match an input datum to a collection of data patterns stored in the CAM array.[9] If this operation occurs in parallel,

then we can obtain high-throughput comparisons with minimal latency, a distinct advantage in applications such as network routing, associative computing, and database management systems.[10–12] In standard silicon-based complementary metal-oxide semiconductor (CMOS) architectures, the construction of a single CAM cell requires approximately 16 transistors, typically configured with static random-access memory (SRAM), leading to significant power consumption and relatively low memory density.[13] NVM has emerged as a promising substitute within CAM, given its reconfigurable functionality, as well as its superior area and energy efficiency. There have already been successful demonstrations of ternary CAM (TCAM) implemented using various NVM technologies.[14–18] However, most CAM designs based on NVM employ designs akin to traditional SRAM, where the memory device is restricted to encoding binary states exclusively and is used for identifying exact matches or mismatches. Given that CAM designs are primarily in the binary domain, current architectures frequently require extensive analog-to-digital conversion (ADC) and digital-to-analog conversion (DAC). ADC and DAC serve as pivotal links between digital memory and analog components (e.g., sensors and actuators), translating various inputs into a form that CAM can handle. The consistent need for conversion significantly diminishes both power and speed, a particularly non-ideal energy overhead in the context of edge computing.[19] More recently, analog CAM (ACAM), which leverages programmable analog conductances within NVM, has been proposed.[20–22] As of now, the hardware development for ACAM poses substantial difficulties and has hence rarely been reported. A recently proposed ACAM cell based on resistive RAM (ReRAM) demands 6 silicon CMOS transistors, resulting in low area efficiency, high parasitic components, and elevated static power consumption.[21] Another proposed ACAM with ferroelectric field-effect transistors (FeFETs) is a theoretical design that requires two search lines (SLs) and a complex analog converter circuit to perform the search signal linear reversion function.[22]

In this work, we report an experimental demonstration of a non-volatile and compact ACAM cell based on two complementary FeFETs. Our ACAM cell utilizes the adjustable threshold voltages of FeFET devices to store over 40 distinct windows within each cell and examines an analog input against this stored range to identify a match or a mismatch. This cell neither requires additional transistors nor an external analog converter circuit—it only requires a single SL. Two key advantages arise from this design: (1) parallel search is conducted where the data resides, eliminating the power and latency costs of data transportation between separate computing and memory units; (2) computation and ADC are merged, offering higher memory bit densities and reduced power usage for a broader set of applications.

We go on to evaluate the performance of our ACAM design for similarity search in a few-shot learning application, using matching networks for inference on the Omniglot dataset and benchmarking both the accuracy and runtime of ACAM against a central processing unit (CPU), graphics processing unit (GPU), and TCAM. Additionally, we can interpret the match operation in ACAM as a kernel and hence deploy ACAM for kernel regression, a model for fitting non-linear functions. Kernels in machine learning are used to model the similarity between two inputs and form the basis for a wide variety of models, including support vector machines (SVMs), kernel regression and classification models, and, most recently, neural tangent kernels for deep learning. Such models typically require intensive computation of pairwise similarity between a test data point and each training data point during inference that could benefit greatly from parallel search.

RESULTS AND DISCUSSION

ACAM cell design and measurement

Different from conventional CAM architectures in which the digital query and keys are compared for identifying exact matches or mismatches (shown in Figure 1A), Figures 1B and 1C illustrate the concept and design of the proposed ACAM, where analog voltage values are supplied as the query to the ACAM and compared against the analog ranges encoded by adjustable threshold voltages of FeFET devices. In this work, the cell structure of ACAM can be significantly simplified by using just two complementary FeFETs (as shown in Figure 1C), in which the n-channel metal-oxide semiconductor (NMOS) FeFET is colored in red, and the p-channel metal-oxide semiconductor (PMOS) FeFET is colored in blue. The analog input search data get converted into voltage amplitudes that are applied along the SL, while the stored analog range is set by the programmed threshold voltages of the cell's two complementary FeFETs. The search operation begins by pre-charging each match line (ML) to a high voltage level. The ML remains high (indicating a match) only when the discharge currents at all ACAM channels are minimal. That is, all attached CAM cells of a row match to the input data. Otherwise, the significant discharge currents of any ACAM channels result in a voltage drop (signifying a mismatch) on the ML. FeFETs work by utilizing positive or negative gate pulses to direct the ferroelectric polarization toward the channel or gate metal, setting the FeFET to a low or high threshold voltage state, respectively. As opposed to other NVM devices that demand substantial DC conduction current for memory write, FeFETs excel in write energy efficiency, as they solely rely on the electric field to switch the polarization. The devices used in this work are FeFETs with hafnium zirconium oxide (HZO) as the ferroelectric gate dielectric integrated on traditional silicon (Si) CMOS transistor technology (see experimental procedures for more details).[23] As a result, these devices can be generalized to any complementary FeFET technology.

We will now discuss how the complementary FeFETs in the ACAM cell store analog match windows to compare against the input search value. To begin, the ML is connected to NMOS FeFET and PMOS FeFET in parallel. Further, the ML stays high for a match outcome when the SL voltage is lower than the threshold voltage of the NMOS FeFET and higher than the threshold voltage of the PMOS FeFET, thus maintaining both FeFET channels in a cutoff state. As shown in Figure 1D, the upper and lower bounds of each match window can be programmed by the threshold voltages of the NMOS FeFET and PMOS FeFET, respectively. When the SL voltage (V_{SL}) exceeds the threshold voltage of the NMOS FeFET, it gets turned on. This leads to a substantial discharge current between the ML and ground (GND), primarily through the NMOS FeFET, which

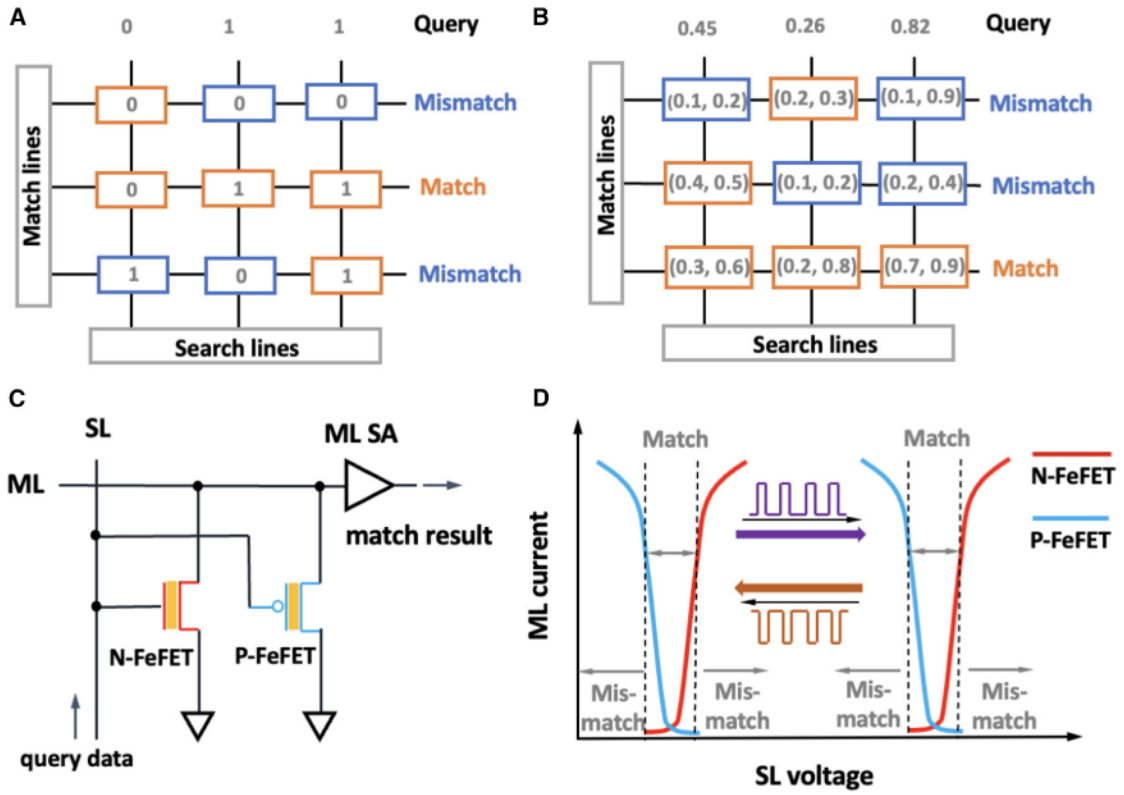


Figure 1. Addressable content-addressable memory (CAM) concept and design

(A) Schematic diagram of digital CAM architectures in which digital query and keys are compared for identifying exact matches or mismatches. Considering that current CAM designs primarily operate in the digital domain, their bit-density and functionality are limited. These architectures often rely heavily on ADC operations to serve as critical interfaces between digital memory and analog components, which significantly reduces both power and speed, a considerable energy overhead, especially in the context of edge computing.

(B) Schematic diagram of analog CAM architecture where analog query is compared with analog intervals, aiming to identify whether it falls within (match) or outside of these intervals (mismatch).

(C) We substantially simplify ACAM cell structure by using just two complementary FeFETs, where the NMOS FeFET is represented in red and PMOS FeFET in blue. Analog input search data are converted into voltage amplitudes and applied along the search line (SL), while stored analog range is determined by the programmed threshold voltages of the two complementary FeFETs within the cell. Similarity between query and keys output on the match line (ML) is shown.

(D) Upper and lower limit of each match window can be programmed by the threshold voltages of NMOS FeFET and PMOS FeFET, respectively. Furthermore, threshold voltage of both NMOS FeFETs and PMOS FeFETs can be adjusted using electrical pulses. This offers an additional level of adaptability to the ACAM cell by providing precise control over FeFETs' properties, thereby enabling the storage of continuous intervals as a match window (for more details, refer to the supplemental information).

in turn causes a voltage drop in the ML that produces a mismatch outcome. On the other hand, when V_{SL} is below the threshold voltage of the PMOS FeFET, it is turned on, leading to a substantial discharge current between the ML and GND, predominantly across the PMOS FeFET. This results in a voltage drop in the ML, also producing a mismatch outcome. When the SL voltage resides within the threshold voltages of both NMOS FeFETs and PMOS FeFETs, both types of FeFETs are in a cutoff state. Consequently, no discharging current will traverse through either of the FeFETs between the ML and GND, and the ML remains at a high level, denoting a match result. In addition, the threshold voltage of both NMOS FeFETs and PMOS FeFETs can be programmed using electrical pulses, as shown in Figure 1D. This allows for precise and efficient tuning of the FeFETs' characteristics, adding an extra layer of flexibility to the ACAM cell and enabling it to store continuous intervals as a match window.

In order to substantiate our circuit design and delve deeper into the concept of complementary FeFET-based ACAM, we perform experimental measurements of ACAM circuit operation on HZO Si CMOS FeFET chips (see Notes S1 and S2, Figure S1). We first present the programmable threshold voltages in both NMOS FeFETs and PMOS FeFETs. These threshold voltages serve to define the upper and lower bounds of the ACAM match window, respectively. We note that we achieve these programmable threshold voltages through partial polarization switching, a process that can be implemented by biasing short electrical pulses at the gate electrode of the FeFETs.[24] Figure 2A illustrates the gradual programming of the I_D-V_G transfer curve in the NMOS FeFET using a series of stepwise gate voltage pulse modulations, ranging from 3 to 4 V. This gradual programming enables the NMOS FeFET to present 10 unique transfer curves, each exhibiting a high level of linearity. Figure 2B demonstrates

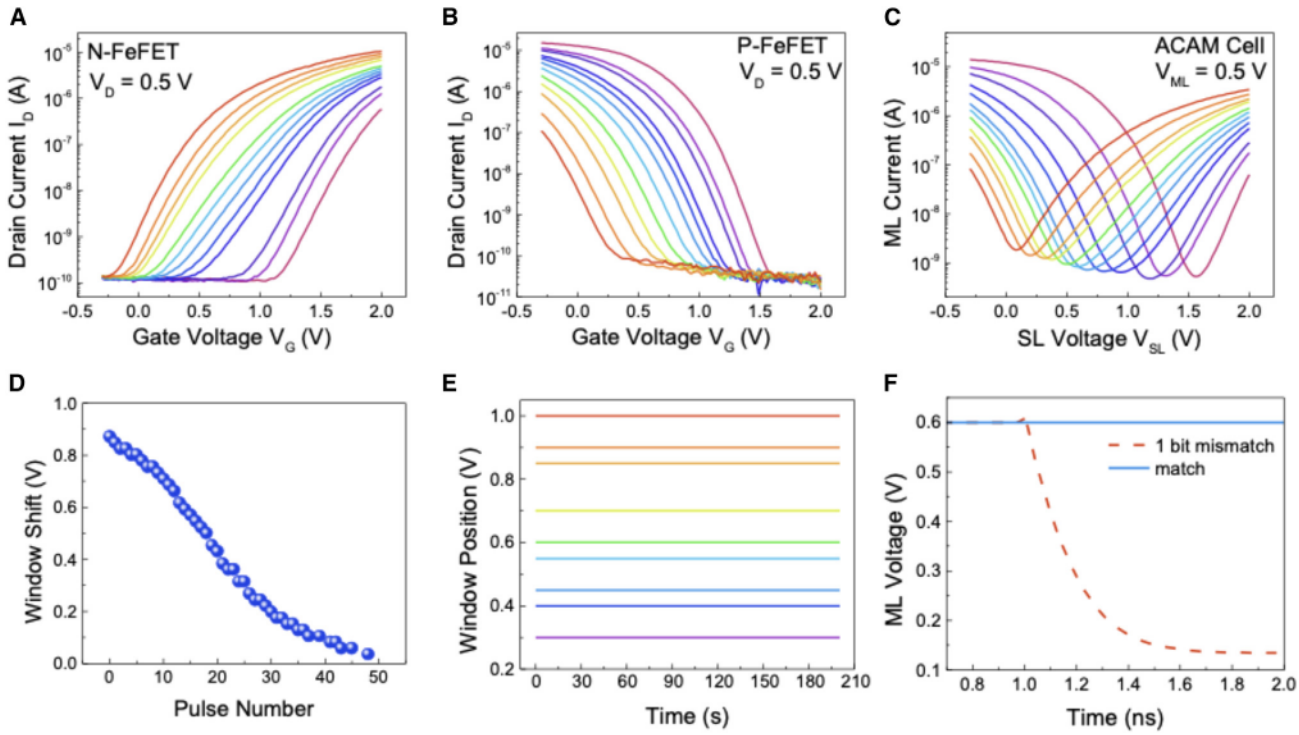


Figure 2. Experimental demonstration of analog content-addressable memory (ACAM) performance

(A) Analog programming of the I_D-V_G transfer curve in NMOS FeFET via stepwise gate voltage pulses of ~ 1 μs, ranging between 3 and 4 V. NMOS FeFET successively yields 10 distinct transfer curves due to this graduated programming.

(B) PMOS FeFET, like its NMOS FeFET counterpart, is also shown to be capable of analog threshold voltage programming through electrical pulses. It is important to note that threshold voltages of both NMOS FeFETs and PMOS FeFETs are influenced in the same manner by applied pulses, as these voltages are contingent on the absolute value of polarization within the ferroelectric dielectric of each device.

(C) Current passing through ML is measured when sweeping voltage is applied on SL. By applying different voltage pulses, the central position of the match window can be adjusted, hence generating multiple match windows at various positions.

(D) Position of ACAM match window can be analogously adjusted, thus providing over 40 distinct match windows.

(E) Demonstration of 9 stored ranges exhibiting stable retention without significant degradation indicates that ACAM is non-volatile.

(F) Simulated ML discharge behavior during search operation, specifically for all-match and 1-bit-mismatch states, in a 64-row, 64-column ACAM array on 45-nm Si CMOS technology. Simulations indicate notable distinction, with an estimated ML delay of 0.136 ns.

that, akin to the NMOS FeFET, the PMOS FeFET is also capable of undergoing analog threshold voltage programming induced by electrical pulses. Positive pulses will make the threshold voltage more positive for both NMOS FeFETs and PMOS FeFETs, as the threshold voltage depends on the absolute value of the amount of polarization in the ferroelectric dielectric. It is noteworthy that the application of a positive pulse in both NMOS FeFETs and PMOS FeFETs elevates their threshold voltages. This phenomenon occurs because these threshold voltages hinge upon the absolute quantity of polarization in the ferroelectric dielectric in both device types.

Subsequently, we perform a proof-of-concept measurement of the performance of the designed ACAM cell. We first define a match window by programming the threshold voltages of the NMOS FeFETs and PMOS FeFETs via electrical pulses over the gate terminal. Then, we measured the ML discharge current with a sweeping SL voltage. As shown in Figure 2C, for each fixed match window, the ML current remains low near the center (signifying a match) and drastically surges as the SL voltage deviates from the window center (signifying a mismatch). Moreover, we observe that by applying voltage pulses, we are able to vary the central position of the match window, thereby creating multiple match windows at different positions. As depicted in Figure 2D, not only is the input in the analog domain, but the location of the ACAM match window can also be analogously programmed, offering more than 40 distinct match windows. This demonstrates the successful operation of the FeFETs in an ACAM cell. As illustrated in Figure 2E, stable retention without noticeable degradation is exhibited for 9 distinct stored ranges, which suggests that ACAM is not only non-volatile but does not require static power for range retention or require frequent refreshing once programmed. For the fast data operations discussed in this paper, a retention period of hundreds of seconds is more than sufficient.[4,11,25] It is also worth noting that much high retention has been observed in our prior works on similar HZO Si CMOS FeFETs.[26] In order to evaluate latency of the search operation in ACAM, we carry out extensive circuit-level analysis of a 64-row, 64-column ACAM array, built upon 45-nm Si CMOS technology and inclusive of peripheral circuits (see Note S3, Figure S2). Figure 2F presents the simulated

ML discharge patterns during a worst-case search operation (considering all-match and 1-bit-mismatch states), clearly demonstrating a significant difference, with the delay in the ML quantified as 0.136 ns. Similar to previous CAM literature, a comparison analysis is provided in Note S8 and Table S1 showing area/bit, search delay, and search energy of our proposed ACAM alongside other designs.[27,28]

ACAM for similarity search

An ACAM cell can store real-valued intervals, as opposed to bits in a binary or ternary memory cell, giving us a new way to search and retrieve directly in the analog domain without converting signals into their digital counterparts. Similarity search is a key problem in machine learning, forming the building block for many applications, such as content retrieval, where a user seeks to recover sentences, audio files, or images that are similar to a given query. Machine learning models, including *k*-nearest neighbor classifiers, SVMs, and kernel machines, have similarity search as a key component of their inference mechanism. A growing body of existing work has therefore been devoted to accelerating these computations.[11,29–31] We show that ACAM cells can be used to store data in their native, real-valued format. Furthermore, compact arrays of such cells can perform similarity searches for a user query at a very low latency, which we will demonstrate for inference in few-shot learning.

As image classification systems begin to tackle more and more classes, the cost of annotating a massive number of images, as well as the difficulty of procuring images of rare categories, increases. This challenge has fueled interest in few-shot learning, a type of learning where only a few labeled samples per class are available for training. The key idea behind current few-shot learning methods is to train a model to distinguish inputs, say images of different categories, from each other. This strategy is different from classical supervised learning, which seeks to predict the category of an input image. Features learned in such models using large datasets, such as the ImageNet-21K dataset (14.2 million images from 21,814 different categories), can be fruitfully used to distinguish between images of entirely new categories using very few new images (i.e., "few-shot").[32] One of the key steps in doing so involves calculating the similarities between the features of few-shot labeled data and the test data, as suggested by an algorithm called "matching networks."[33] For *k*-shot learning across *n* different classes (typically, *k* ranges from 1 to 10, and *n* can range anywhere from 5 to 100), the centroid of the features of the *k* images of each class is computed. Given a test image (also called the "query"), the similarity (e.g., Hamming distance, inner product) of the features of the test image, which could be from any of the *n* classes, is computed to find the centroid closest to it. See the schematic in Figure 3A for an example where the unknown query sample (i.e., a Dachshund breed) should be matched to the Yellow Labrador breed, as they are both dogs, in the 4-way, 1-shot support set containing a dog, cat, pig, and fish.

Similarity search in few-shot learning can be mapped to ACAM as follows. In software, the support embeddings are stored in an array of size $n \cdot k \times 64$, while the query sample is represented as a 64-dimensional array. Similarly, an $(n \cdot k \times 64)$-dimensional ACAM array is used to store the features of few-shot labeled images embeddings, where the (i,j) th element corresponds to the j th element of i th support embedding. Programming of the support embeddings into ACAM involves selecting a window size, typically held constant across all ACAM cells, and then positioning the window such that the (i,j) th element lies at the center of its window for all $i = 1, \ldots, n \cdot k$ and $j = 1, \ldots, 64$. Recall that the central position of the match window can be adjusted by gradually applying voltage pulses to the NMOS FeFET or PMOS FeFET, pictured in each ACAM cell of the Figure 3A circuit diagram. On the other hand, the query embedding is programmed onto the SLs, shown at the top of each column in the Figure 3A circuit diagram, such that the j th element of the query embedding lies on the j th SL of the $n \cdot k \times 64$ ACAM array for all $j = 1, \ldots, 64$. Binary and ternary CAM compute conventional Hamming distance, which checks for exact match between bits and can be written as $\boldsymbol{S}_{\text{digital}}(\mathbf{s}_i, \mathbf{q}) = \sum_{j=1}^{64} \mathbf{1}(\mathbf{s}_i[j] = \mathbf{q}[j])$, where $\mathbf{1}(\cdot)$ is the indicator function. In contrast, ACAM computes a "generalized" Hamming distance within the analog domain, which can be expressed as $\boldsymbol{S}_{\text{analog}}(\mathbf{s}_i, \mathbf{q}) = \sum_{j=1}^{64} \mathbf{1}(\mathbf{q}[j] \in [a_{i,j}, b_{i,j}])$, where $a_{i,j}$ and $b_{i,j}$ are the endpoints of the match window in the ACAM cell corresponding to the j th element of i th support embedding. To reiterate, the main advantage of ACAM over binary and ternary CAM is that it computes similarity directly in the analog domain via generalized Hamming distance.

Selection of the matching window size and impact of added noise within ACAM will now be discussed in the context of similarity search. Figures 3B and 3C are benchmarked with a 5-way, 5-shot inference task on the Omniglot dataset, which contains 1,623 characters from 50 different alphabets and is downsampled such that each image is 28 × 28. Further, all support embeddings stored in ACAM are quantized to lie within the range [–0.3, 2.0] V to align with the range of operation shown in Figures 2A–2C, falling within the memory window of the current FeFET device. When programming the support embeddings into ACAM, the first step is to select a window size. A match window range of within [0.0, 1.0] V is sufficiently expressive to achieve the highest inference accuracy, which occurs at a match window size of 0.4 V as seen in Figure 3B. While it is not possible to avoid noise within CAM, it is nonetheless possible to show that ACAM is robust to any perturbations in its match window when used for similarity search in few-shot learning. The impact of variation in the FeFET device/ACAM on the accuracy of similarity search task should be carefully examined. The predominant variation in FeFET devices originates from phase and grain changes in ferroelectric HfO_2.[34–38] It is possible to attain a maximum standard deviation of less than 100 mV for the discrepancy between the actual and target V_{th} in the reported FeFET devices.[39] Figure 3C shows the result of noise sampled from a normal distribution with mean $\mu = 0$ and standard deviation σ being added to the match window, illustrating that a window variation of up to ± 0.1 V, which is ± 4.35% of the entire range of ACAM operation, is tolerable without incurring a decrease in inference accuracy of more than 10%. Additionally, Note S7 and Figure S5 show that ACAM is robust to cycle-to-cycle variation of up to 10% of the match window size for inference on a 20-way, 1-shot task.

It is important to compare the density of ACAM versus TCAM. Analog representations in ACAM are far more expressive than the digital representations in TCAM, which in turn yields denser

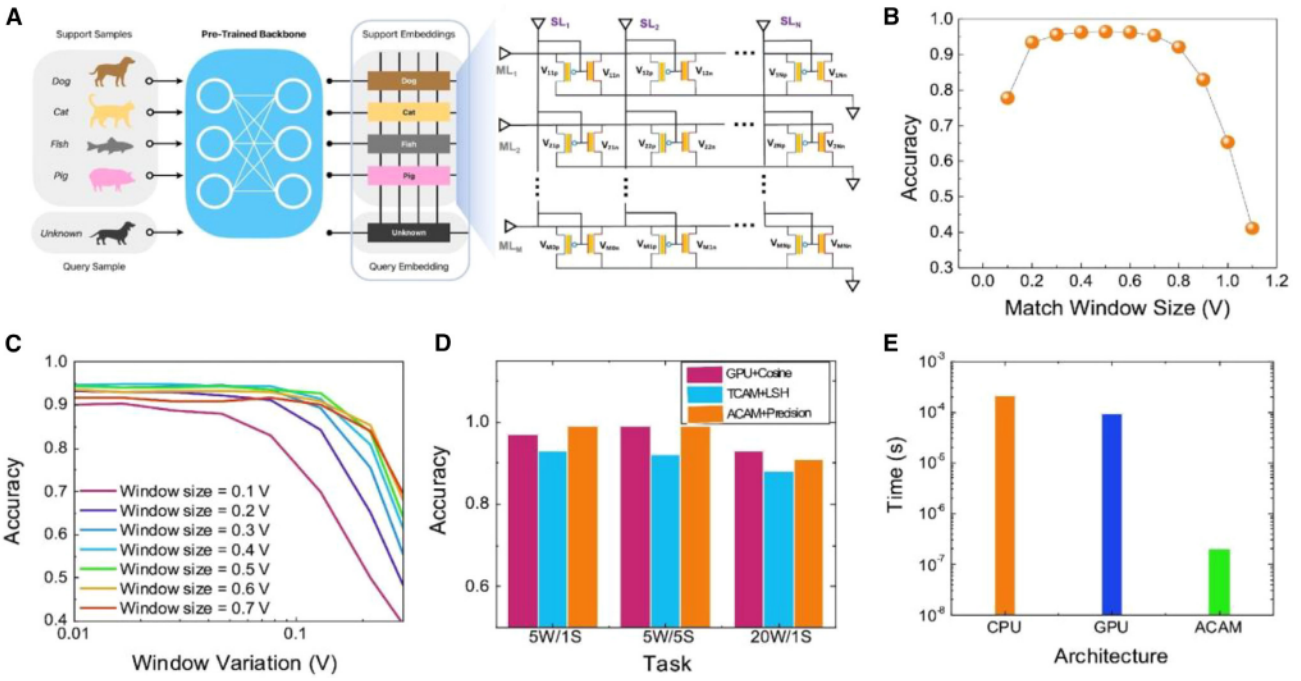


Figure 3. Matching networks inference for few-shot learning on analog content-addressable-memory (ACAM)

(A) 4-way, 1-shot few-shot learning episode with dog, cat, fish, and pig encoded as 64-dimension support embedding by pre-trained backbone. Each element of support embedding is stored in the ACAM cell, while query embedding is placed on the search line. Match line outputs 4-dimensional vector scoring similarity between query embedding and 4 support embeddings, with query label given as label of support sample with largest similarity score.

(B–E) Omniglot dataset partitioned into *n*-way, *k*-shot episodes. A 4-layer convolutional neural network, called the backbone, outputs 64-dimensional embedding for each sample. Each layer in the backbone has a 3 × 3 kernel with 64 filters, batch normalization, ReLU non-linearity, and a 2 × 2 max-pooling. Backbone is trained on 1,200 characters, while inference occurs on the remaining 423 characters.

(B) Optimal match window size is shown to be 0.4 V in order to maximize inference accuracy.

(C) Inference accuracy remains approximately constant at 90%–95% for match window size ranging between 0.2 and 0.7 V for window variation up to 0.1 V.

(D) Benchmark of inference accuracy shows ACAM outperforms TCAM for 5-way and 20-way tasks and remains comparable to GPU. TCAM+LSH data were sourced from Ni et al.[11] which use a 28-nm CMOS node. But ACAM is a notably denser memory architecture, requiring only 64 cells per 64-dimensional embedding, while TCAM needs on the order of 128 or 256 cells.

(E) Amount of time in seconds to perform inference on ACAM (45-nm node) for one query sample shown to be more than 2 orders of magnitude less than both CPU (14-nm node) and GPU (12-nm node).

memory storage while maintaining high-accuracy results. Specifically, binary/ternary representations require the use of locality-sensitive binary codes. It is shown that binary classification on 14,871 images of dimension 320 taken from the LabelMe database result in a diverse set of precision-recall curves when locality-sensitive binary codes are used to embed the test data with various code sizes.[40] Precision represents the fraction of positive predictions that actually belong to the positive class, while recall is the fraction of positive predictions out of all positive instances in the dataset. The goal is to maximize both. For a given recall value of 0.8, the precision varies between 0.4, 0.65, and 0.8 for 256-, 512-, and 1,024-bit codes, respectively. It is observed that larger bit codes (e.g., 512 or 1,024) are more suitable for real-world applications. Within matching networks, the backbone embeds 784-dimension Omniglot data into 64 dimensions. If 512- or 1,024-bit codes are strongly suggested for 320-dimensional data, it is reasonable to expect at least 128- or 256-bit codes for the Omniglot dataset, which relates to an $n\times128$ or $n\times256$ TCAM array. In sharp contrast, ACAM requires only an $n\times64$ array, meaning ACAM is a 3× denser memory architecture than TCAM. A detailed explanation for the density enhancement has been presented in Note S4 and Figure S3.

Lastly, the performance of matching networks with ACAM will be benchmarked with respect to accuracy and runtime. After executing inference on 423 characters, Figure 3D shows the accuracy results of ACAM (45-nm node) are on average 5% higher than those of TCAM (28-nm node) and comparable to those of the GPU (NVIDIA Tesla T4, 12-nm node), where TCAM+LSH data were sourced from Ni et al.[11] Additionally, the time incurred for a single inference operation in ACAM is significantly less than both CPU (Intel Xeon Platinum, 14-nm node) and GPU by more than 2 orders of magnitude, as shown in Figure 3E. To summarize, ACAM represents an efficient alternative to CPU, GPU, and TCAM for similarity search in few-shot learning that not only outperforms TCAM in its inference results (while remaining comparable to the GPU) but does so with a denser memory architecture that enables faster, parallel computation. It is further worth noting that the above is a conservative estimate since the ACAM is computed on a 45-nm Si CMOS mode, with Si FeFETs currently available in the more advanced 28-nm

technology node, while the CPU and GPU results follows from 14-nm to 12-nm nodes, respectfully.[11,23] Further details on the software simulation can be found in Note S5.

ACAM for kernel regression

We next demonstrate how to use ACAM for inference in a kernel regression machine.[41] Given two inputs $\mathbf{x}$ and $\mathbf{x}'$, which are d-dimensional vectors, a kernel is a function $K(\mathbf{x},\mathbf{x}')$ that can be understood as an estimate of the similarity between the two inputs. Such a function can be used to make predictions as follows. Given a training dataset $\mathcal{D}_{\mathrm{Tr}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$, where $\mathbf{x}_i$ are the inputs and $y_i \in \mathbb{R}$ are the outputs, we compute the "Gram matrix" $\mathbf{K}[i,j] = K(\mathbf{x}_i, \mathbf{x}_j)$ whose entries are the pairwise similarities between inputs in the training set. Predictions on a new test datum $\mathbf{x}$ are computed as $\widehat{y}(\mathbf{x}) = \sum_{i=1}^{m} \widehat{\alpha}_i K(\mathbf{x}_i, \mathbf{x})$, where $\widehat{\alpha} = (\mathbf{K}+\lambda m \mathbf{I}_m)^{-1}\mathbf{y}$ is the vector of coefficients that are used to weigh the true targets of the m samples, denoted by $\mathbf{y}$, to make the predictions on the test datum $\mathbf{x}$. Targets of input samples that are more similar to the test datum are up-weighted in the above summation. Note that the parameter λ is known as the ridge regression constant, allowing us to regularize the fitting procedure in situations when there are few samples in the training dataset. Observe that it biases the diagonal of the Gram matrix $\mathbf{K}$ and effectively makes each input sample play a role in making predictions on the test datum. Mathematical details of this procedure are provided in Note S6.

There are many kernels $K(\mathbf{x},\mathbf{x}')$ that can be used in this procedure. One popular choice involves a radial basis function kernel $K(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x}-\mathbf{x}'\|_2^2 / 2\gamma^2)$, which computes the probability of the test datum being drawn from a Gaussian distribution centered at one of the training data points, as seen in Figure 4AI. Another version consists of the Laplace kernel $K(\mathbf{x},\mathbf{x}') = \exp(-c\|\mathbf{x}-\mathbf{x}'\|)$. Constants like γ and c in these expressions are considered hyperparameters and are chosen using cross-validation. Computing the prediction in a kernel regression machine involves computing the summation above, and even if coefficients $\widehat{\alpha}$ are computed beforehand, it is necessary to compute the m terms $K(\mathbf{x}_i, \mathbf{x})$ for each new test datum $\mathbf{x}$. Further, all the m training samples must be stored in memory.

ACAM offers a different way to implement kernel regression. Observe that the current-voltage characteristic curve in an ACAM computes a kernel. The transfer curves shown in Figure 2C can be approximately described by the expression $\exp(|V_G - \mu|^2 / 2\gamma^2)$, where V_G is the applied gate voltage, and μ is the mean of the match window. This expression can be equivalently written using the previous kernel notation as $\exp(-|x-x'|^2 / 2\gamma^2)$. Since this expression looks like a parabola, a "surrogate" Gaussian kernel is achieved by the following constant-time negation, summation, and maximization operations, yielding $K^{\mathrm{ACAM}}(\mathbf{x}, \mathbf{x}') = \max\{0, 2 - \exp(|\mathbf{x}-\mathbf{x}'|^2/2\gamma^2)\}$. See Figure S4A for an overlaid plot of the Gaussian kernel and surrogate Gaussian kernel. This surrogate kernel is what ACAM outputs on the ML for some stored vector $\mathbf{x}$ and query vector $\mathbf{x}'$.

It will be shown that ACAM enables rapid, 1-step inference. All that is required is for a test data point to be placed on the SLs of the ACAM. The surrogate Gaussian kernel can be computed in *O(1)* time. Figure 4Aii shows how the multiplication with $\widehat{\alpha}$ can also be designed to occur within the same step by placing $\widehat{\alpha}_i$ on the drain of both the NMOS FeFET and PMOS FeFET in the i th ACAM cell for all $i = 1,\ldots,m$, which linearly scales each output $K(\mathbf{x}_i, \mathbf{x}')$ by $\widehat{\alpha}_i$. Since the ML sums input currents, the ML output becomes $\sum_{i=1}^{m} \widehat{\alpha}_i K(\mathbf{x}_i, \mathbf{x}')$, the predicted label $\widehat{f}(\mathbf{x}')$ under kernel regression. Hence, ACAM is able to achieve 1-step inference for kernel regression.

To evaluate the ability of ACAM to perform such inference, 1-dimensional synthetic data were generated by randomly sampling $f(x) = \sin(5x)$, known as the ground-truth function, and adding noise randomly sampled from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 0.2$. This process is then used to generate both the training and test data. Like in the case of few-shot learning, the ACAM-based kernel regression model exhibits robustness to noise. As seen in Figure 4B, even with a window variation of up to 0.3 V, the mean squared error (MSE) of the fitted function only changes on the order of 10^{-3} to 10^{-2}. Furthermore, Figure 4B shows that the reduction in MSE saturates at 4 bits, after which additional bits do not lead to diminished inference error. Since ACAM operates at about 4 bits per cell, Figure 4B suggests that neither quantization nor noise significantly degrade the performance of ACAM-based inference.

Benchmarking of the time required to complete inference on a single test data point shows that ACAM (45-nm node), performs 3 orders of magnitude faster than CPU and GPU (12-nm nodes) similar to the above demonstration of few-shot learning. See the supplemental information for a justification on why CPU and GPU have been placed in the same column. Furthermore, Figure 4C shows that both CPU and GPU perform on the order of $64 \cdot 64 = 4,096$ floating-point operations (FLOPs) during inference, while ACAM requires only 1.

Figures 4Di–4Diii illustrate the predictions (red) made by ACAM on the scattered test data points (blue). The quality of the fit can be observed by the closeness between the ground-truth function (black) and the ACAM predictions. Note that the surrogate Gaussian kernel has a parameter γ that determines the width of the function and, by extension, the slope of the current-voltage plot. From Figure 2C, this value can be approximated as $\gamma = 0.1$ V. Figures 4Di and 4Diii show the test predictions for $\gamma > 0.1$ V, $\gamma = 0.1$ V, and $\gamma < 0.1$ V, corresponding to underfitting, optimal, and overfitting conditions, respectively, and Figures S4B and S4C show how training and test data can be scaled in order to ensure that $\gamma = 0.1$ V remains the optimal condition. To further evaluate the impact of noise, we determine the predictions outputted by ACAM under noisy conditions $\mathbf{y}_{\mathrm{noise}}$. After that, we compute the residuals $\mathbf{y}_{\mathrm{noise}} - \mathbf{y}_{\mathrm{ground-truth}}$, which quantify how close the predicted labels are to the ground-truth labels and plot them as a histogram in Figures 4Div–4Dvi. As expected, the histogram traces out a Gaussian distribution with mean $\mu = 0$ for a window variation of 0.01 V, meaning the vast majority of predicted labels were the same as the ground-truth labels (shown in Figure 4Div). For a window variation of 0.08 V, it can be seen that the fitted Gaussian is no longer quite centered at mean $\mu = 0$, while the variance has increased (shown in Figure 4Dv). The trend in the means and variances of Gaussian functions fitted to the residuals as noise increases can be seen in Figure 4Cvi. Though the variances

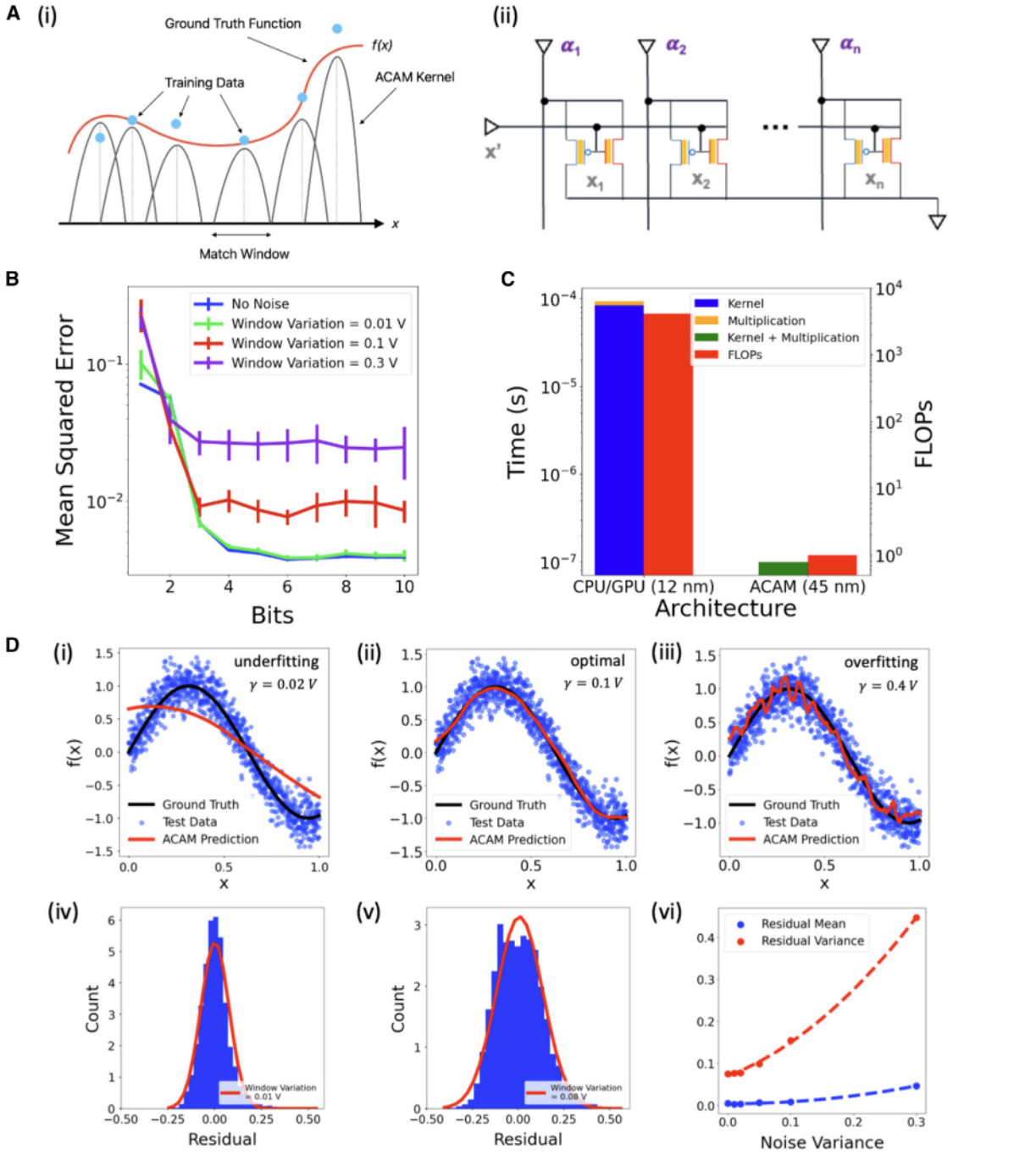


Figure 4. Inference for kernel regression model on analog content-addressable-memory (ACAM)

(A) (i) Fitting kernel regression model can be thought of as summing a set of Gaussian functions, with mean centered at each data point, to get a function that reflects all the given data. (ii) Circuit that computes the predicted label $\hat{y}(\mathbf{x}) = \sum_{i=1}^{m} \hat{\alpha}_i K^{\mathrm{ACAM}}(\mathbf{x}_i, \mathbf{x})$ during the inference phase in 1 step.

(B) Mean squared error (MSE) remains less than 0.03 for 4-bit quantization, approximately what ACAM operates at, with window variation up to 0.3 V.

(C) Benchmark of CPU and GPU (12-nm node) versus ACAM (45-nm node) in terms of time in seconds and number of floating-point operations (FLOPs) needed to perform inference on a single test datum, with ACAM again outperforming its counterparts by about 3 orders of magnitude.

(D) (i) Underfitting exhibited by ACAM predictions (red) on test data (blue) with too large of a kernel parameter $\gamma = 0.4$ V. (ii) Optimal fitting exhibited by ACAM predictions (red) on test data (blue) with just right of a kernel parameter $\gamma = 0.1$ V, the typical width of ACAM surrogate Gaussian kernel. (iii) Overfitting exhibited by ACAM predictions (red) on test data (blue) with too small of a kernel parameter $\gamma = 0.02$ V. (iv) Histogram of residual $\mathbf{y}_{\mathrm{noise}} - \mathbf{y}_{\mathrm{ground-truth}}$ for small amount of Gaussian noise ($\mu = 0, \sigma = 0.01$) added to ACAM cells. (v) Histogram of residual $\mathbf{y}_{\mathrm{noise}} - \mathbf{y}_{\mathrm{ground-truth}}$ for moderate amount of Gaussian noise ($\mu = 0, \sigma = 0.08$) added to ACAM cells. (vi) Trend of means and variances of Gaussian fit to residual histogram as noise increases, showing that ACAM boasts robustness to noise, as residual means remains close to $\mu = 0$.

increase quadratically, the means remain close to $\mu = 0$, which further confirms that ACAM remains tolerant to noise in inference. Note S7 and Figure S5 once again confirm that ACAM can tolerate cycle-to-cycle variation of up to 10% of the match window size for inference on a kernel regression task, with data sampled from the sine function. To recapitulate, this section has presented a 1-step inference procedure for kernel regression that leverages the strengths of ACAM in computing a (surrogate) Gaussian kernel. This result extends previous work of CIM hardware for linear regression to non-linear regression and opens the door to future applications of ACAM in other attention- or kernel-based learning.[42]

Conclusion

In summary, we demonstrate an ACAM CIM architecture based on complementary FeFETs. We experimentally validate the operation and function of individual ACAM cells on HZO Si complementary FeFET devices and then use simulation tools to compare ACAM performance in similarity search and kernel regression. We demonstrate that our ACAM architecture at 45-nm CMOS node can outperform CPU and GPU at 12-nm CMOS node in similarity search by 2 orders of magnitude and by 3 in kernel regression. Given these advantages of our ACAM CIM over CPU and GPU, the question becomes what kind of CIM-based pattern matching architecture to use. Among CAMs, the choice is between digital TCAM or our presented ACAM. TCAM requires extensive ADC and DAC operations that significantly reduce both power and speed, providing a limited advantage in pattern matching performance over algorithms on CPU and GPU. As a result, the proposed ACAM in this work represents a strong candidate for accelerating pattern matching in machine learning. In the future, CAM-based pattern matching architectures could be mapped to other machine learning or robotics tasks, including visual scene understanding, kernel SVMs, or even attention in large language models, like transformers.

EXPERIMENTAL PROCEDURES

Resource availability

Lead contact

Further information and requests for resources should be directed to the lead contact, Deep Jariwala (dmj@seas.upenn.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The data and code that support the conclusions of this study are also available from the lead contacts upon reasonable request.

Device fabrication

NMOS FeFETs and PMOS FeFETs were fabricated in the Rochester Institute of Technology (RIT) student-run fabrication facility on 1–10 $\Omega \cdot$ cm base resistivity silicon wafers using the RIT CMOS process consisting of LOCOS isolated field-effect transistors with ion-implanted source and drain regions. Custom masks were designed using Mentor Graphics Pyxis and fabricated using the Heidelberg DWL 66+ laser writer. Photolithography was performed using an i-line ASML PAS 5500/200 stepper. Ferroelectric gate dielectric and gate metal stack deposition was performed at Namlab, Germany, with 11-nm-thick $Hf_{0.5}Zr_{0.5}O_2$ (HZO) films deposited via atomic layer deposition by alternating cycles of HfO_2 and ZrO_2 using $HfCp(NMe_2)_3$ and $ZrCp(NMe_2)_3$ as metal-organic precursors and ozone as an oxidant on a native SiO_2 layer on Si. A TiN top electrode was deposited via sputtering under ultra-high vacuum. Both films were annealed at 500°C for 20 s in N_2. Measurements on a metal-ferroelectric-metal capacitor structure fabricated in the same deposition run as the FeFET devices showed remanent polarization (P_r) values of about 20 $\mu C/cm^2$.

Device characterization and simulation

Current-voltage measurements were performed in air at ambient temperature using a Keithley 4200A semiconductor characterization system. The ACAM array inference is simulated using SPICE simulation. This simulation fully incorporates the ACAM array, under the assumption that the peripheral circuits, inclusive of the ML sense amplifier (see Figure S2), are grounded in 45-nm silicon CMOS technology. In terms of inference applications, the focus is solely on benchmarking the search operation. Given that inference does not include a write operation, the characteristics of the ACAM cell are simulated using parameters from 45-nm silicon CMOS technology transistors.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.device.2023.100218.

ACKNOWLEDGMENTS

K.K. acknowledges support from the National Science Foundation (NSF) Graduate Research Fellowship Program (GRFP), Fellow ID: 2022338725. D.J. and X.L. acknowledge partial support from Intel Rising Star Faculty Award. P.C. was supported by grants from the NSF (IIS-2145164, CCF-2212519), the DARPA Quantum Inspired Classical Computing (QUICC) program, and an Intel Rising Star Faculty Award. U.S. was financially supported out of the Saxonian State budget approved by the delegates of the Saxon State Parliament. S.K. acknowledges the HZO Si FE-FET device development work originated from the Semiconductor Research Corporation (SRC) funding received under the SRC Global Research Collaboration (GRC) task 2825.001. The NSF, Grant #EEC-2123863, also supported part of this collaborative work. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF. The authors thank Roy H. Olsson at UPenn for allowing them access to probe station in his lab for electrical measurements.

AUTHOR CONTRIBUTIONS

X.L., K.K., D.J., and P.C. conceived the idea. X.L. conceptualized and designed the FeFET-based ACAM circuits and in-memory search architecture. X.L. and K.K. performed machine learning simulations. X.L. and Y.H. performed electrical measurements and SPICE simulations. P.J. performed microfabrication of FeFETs under supervision of S.K., U.S., and C.R. X.L., K.K., D.J., and P.C. analyzed and interpreted the data. All authors contributed to writing of the manuscript.

DECLARATION OF INTERESTS

X.L., K.K., D.J., and P.C. have filed a provisional patent application based on the idea.

Received: September 11, 2023
Revised: November 1, 2023
Accepted: November 22, 2023
Published: January 19, 2024

REFERENCES

1. Wong, H.S.P., and Salahuddin, S. (2015). Memory leads the way to better computing. Nat. Nanotechnol. *10*, 191–194.
2. Ielmini, D., and Wong, H.S.P. (2018). In-memory computing with resistive switching devices. Nat. Electron. *1*, 333–343.

3. Sebastian, A., Le Gallo, M., Khaddam-Aljameh, R., and Eleftheriou, E. (2020). Memory devices and applications for in-memory computing. Nat. Nanotechnol. *15*, 529–544.

4. Yao, P., Wu, H., Gao, B., Tang, J., Zhang, Q., Zhang, W., Yang, J.J., and Qian, H. (2020). Fully hardware-implemented memristor convolutional neural network. Nature *577*, 641–646.

5. Li, C., Belkin, D., Li, Y., Yan, P., Hu, M., Ge, N., Jiang, H., Montgomery, E., Lin, P., Wang, Z., et al. (2018). Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. Nat. Commun. *9*, 2385.

6. Ambrogio, S., Narayanan, P., Tsai, H., Shelby, R.M., Boybat, I., di Nolfo, C., Sidler, S., Giordano, M., Bodini, M., Farinha, N.C.P., et al. (2018). Equivalent-accuracy accelerated neural-network training using analogue memory. Nature *558*, 60–67.

7. Berdan, R., Marukame, T., Ota, K., Yamaguchi, M., Saitoh, M., Fujii, S., Deguchi, J., and Nishi, Y. (2020). Low-power linear computation using nonlinear ferroelectric tunnel junction memristors. Nat. Electron. *3*, 259–266.

8. Zha, Y., Nowak, E., and Li, J. (2020). Liquid Silicon: A Nonvolatile Fully Programmable Processing-in-Memory Processor With Monolithically Integrated ReRAM. IEEE J. Solid State Circ. *55*, 908–919.

9. Pagiamtzis, K., and Sheikholeslami, A. (2006). Content-addressable memory (CAM) circuits and architectures: a tutorial and survey. IEEE J. Solid State Circ. *41*, 712–727.

10. Ravikumar, V.C., and Mahapatra, R.N. (2004). TCAM architecture for IP lookup using prefix properties. IEEE Micro *24*, 60–69.

11. Ni, K., Yin, X., Laguna, A.F., Joshi, S., Dünkel, S., Trentzsch, M., Müller, J., Beyer, S., Niemier, M., Hu, X.S., and Datta, S. (2019). Ferroelectric ternary content-addressable memory for one-shot learning. Nat. Electron. *2*, 521–529.

12. Kai, Z., Che, H., Zhijun, W., Bin, L., and Xin, Z. (2006). DPPC-RE: TCAM-based distributed parallel packet classification with range encoding. IEEE Trans. Comput. *55*, 947–961.

13. Nii, K., Amano, T., Watanabe, N., Yamawaki, M., Yoshinaga, K., Wada, M., and Hayashi, I. (2014). 13.6 A 28nm 400MHz 4-parallel 1.6Gsearch/s 80Mb ternary CAM. In 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 240–241.

14. Liu, X., Ting, J., He, Y., Fiagbenu, M.M.A., Zheng, J., Wang, D., Frost, J., Musavigharavi, P., Esteves, G., Kisslinger, K., et al. (2022). Reconfigurable Compute-In-Memory on Field-Programmable Ferroelectric Diodes. Nano Lett. *22*, 7690–7698.

15. Yang, R., Li, H., Smithe, K.K.H., Kim, T.R., Okabe, K., Pop, E., Fan, J.A., and Wong, H.S.P. (2019). Ternary content-addressable memory with MoS2 transistors for massively parallel data search. Nat. Electron. *2*, 108–114.

16. Song, B., Na, T., Kim, J.P., Kang, S.H., and Jung, S.O. (2017). A 10T-4MTJ Nonvolatile Ternary CAM Cell for Reliable Search Operation and a Compact Area. IEEE Trans. Circuits Syst. II. *64*, 700–704.

17. Fedorov, V.V., Abusultan, M., and Khatri, S.P. (2014). An area-efficient Ternary CAM design using floating gate transistors. In 2014 IEEE 32nd International Conference on Computer Design (ICCD) (IEEE), pp. 55–60.

18. Li, J., Montoye, R.K., Ishii, M., and Chang, L. (2014). 1 Mb 0.41 μm^2 2T-2R Cell Nonvolatile TCAM With Two-Bit Encoding and Clocked Self-Referenced Sensing. IEEE J. Solid State Circ. *49*, 896–907.

19. Shafiee, A., Nag, A., Muralimanohar, N., Balasubramonian, R., Strachan, J.P., Hu, M., Williams, R.S., and Srikumar, V. (2016). ISAAC: a convolutional neural network accelerator with in-situ analog arithmetic in crossbars. In Proceedings of the 43rd International Symposium on Computer Architecture (IEEE Press).

20. Blyth, T.A., and Orlando, R.V. (2006). Analog Content Addressable Memory (CAM) Employing Analog Nonvolatile Storage. patent US 6,985,372 B1.

21. Li, C., Graves, C.E., Sheng, X., Miller, D., Foltin, M., Pedretti, G., and Strachan, J.P. (2020). Analog content-addressable memories with memristors. Nat. Commun. *11*, 1638.

22. Yin, X., Li, C., Huang, Q., Zhang, L., Niemier, M., Hu, X.S., Zhuo, C., and Ni, K. (2020). FeCAM: A Universal Compact Digital and Analog Content Addressable Memory Using Ferroelectric. IEEE Trans. Electron. Dev. *67*, 2785–2792.

23. Ali, T., Polakowski, P., Riedel, S., Büttner, T., Kämpfe, T., Rudolph, M., Pätzold, B., Seidel, K., Löhr, D., Hoffmann, R., et al. (2018). Silicon doped hafnium oxide (HSO) and hafnium zirconium oxide (HZO) based FeFET: A material relation to device physics. Appl. Phys. Lett. *112*, 222903.

24. Jerry, M., Chen, P.Y., Zhang, J., Sharma, P., Ni, K., Yu, S., and Datta, S. (2017). Ferroelectric FET analog synapse for acceleration of deep neural network training. In 2017 IEEE international electron devices meeting (IEDM) (IEEE), pp. 6.2.1–6.2.4.

25. Wan, W., Kubendran, R., Schaefer, C., Eryilmaz, S.B., Zhang, W., Wu, D., Deiss, S., Raina, P., Qian, H., Gao, B., et al. (2022). A compute-in-memory chip based on resistive random-access memory. Nature *608*, 504–512.

26. Yurchuk, E., Müller, J., Müller, S., Paul, J., Pešić, M., van Bentum, R., Schroeder, U., and Mikolajick, T. (2016). Charge-Trapping Phenomena in HfO2-Based FeFET-Type Nonvolatile Memories. IEEE Trans. Electron. Dev. *63*, 3501–3507.

27. Kazemi, A., Sharifi, M.M., Laguna, A.F., Müller, F., Yin, X., Kämpfe, T., Niemier, M., and Hu, X.S. (2022). FeFET Multi-Bit Content-Addressable Memories for In-Memory Nearest Neighbor Search. IEEE Trans. Comput. *71*, 2565–2576.

28. Rajaei, R., Sharifi, M.M., Kazemi, A., Niemier, M., and Hu, X.S. (2021). Compact Single-Phase-Search Multistate Content-Addressable Memory Design Using One FeFET/Cell. IEEE Trans. Electron. Dev. *68*, 109–117.

29. Kumar, P., Nair, A.R., Chatterjee, O., Paul, T., Ghosh, A., Chakrabartty, S., and Thakur, C.S. (2019). Neuromorphic In-Memory Computing Framework using Memtransistor Cross-bar based Support Vector Machines *4–7*, 311–314. . (Accessed August 2019).

30. Jiang, Y., Kang, J., and Wang, X. (2017). RRAM-based parallel computing architecture using k-nearest neighbor classification for pattern recognition. Sci. Rep. *7*, 45233.

31. Mao, R., Wen, B., Kazemi, A., Zhao, Y., Laguna, A.F., Lin, R., Wong, N., Niemier, M., Hu, X.S., Sheng, X., et al. (2022). Experimentally validated memristive memory augmented neural network with efficient hashing and similarity search. Nat. Commun. *13*, 6284.

32. Dhillon, G.S., Chaudhari, P., Ravichandran, A., and Soatto, S. (2019). A baseline for few-shot image classification. Preprint at arXiv.

33. Vinyals, O., Blundell, C., Lillicrap, T.P., Kavukcuoglu, K., and Wierstra, D. (2016). Matching Networks for One Shot Learning. Adv. Neural Inf. Process. Syst., 3637–3645.

34. Choe, G., and Yu, S. (2021). Variability Study of Ferroelectric Field-Effect Transistors Towards 7nm Technology Node. IEEE J. Electron Devices Soc. *9*, 1131–1136.

35. Mueller, J., Slesazeck, S., and Mikolajick, T. (2019). Chapter 10.4 - Ferroelectric Field Effect Transistor. In Ferroelectricity in Doped Hafnium Oxide: Materials, Properties and Devices, U. Schroeder, C.S. Hwang, and H. Funakubo, eds. (Woodhead Publishing)), pp. 451–471.

36. Grimley, E.D., Schenk, T., Mikolajick, T., Schroeder, U., and LeBeau, J.M. (2018). Atomic Structure of Domain and Interphase Boundaries in Ferroelectric HfO2. Adv. Mater. Interfac. *5*, 1701258.

37. Grimley, E.D., Schenk, T., Sang, X., Pešić, M., Schroeder, U., Mikolajick, T., and LeBeau, J.M. (2016). Structural Changes Underlying Field-Cycling Phenomena in Ferroelectric HfO2 Thin Films. Adv. Electron. Mater. *2*, 1600173.

38. Ni, K., Gupta, A., Prakash, O., Thomann, S., Hu, X.S., and Amrouch, H. (2020). Impact of Extrinsic Variation Sources on the Device-to-Device Variation in Ferroelectric FET. In 2020 IEEE International Reliability Physics Symposium (IRPS) (IEEE Press).

39. Soliman, T., Chatterjee, S., Laleni, N., Müller, F., Kirchner, T., Wehn, N., Kämpfe, T., Chauhan, Y.S., and Amrouch, H. (2023). First demonstration of in-memory computing crossbar using multi-level Cell FeFET. Nat. Commun. *14*, 6348.

40. Raginsky, M., and Lazebnik, S. (2009). Locality-Sensitive Binary Codes from Shift-Invariant Kernels. Adv. Neural Inf. Process. Syst., 1509–1517.

41. Schölkopf, B., and Smola, A.J. (2018). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond (The MIT Press).

42. Sun, Z., Pedretti, G., Bricalli, A., and Ielmini, D. (2020). One-step regression and classification with cross-point resistive memory arrays. Sci. Adv. *6*, eaay2378.