chips

MDPI

Review

Approximate Content-Addressable Memories: A Review

Esteban Garzón [1,*], Leonid Yavits [2], Adam Teman [2] and Marco Lanuzza [1]

[1] Department of Computer Engineering, Modeling, Electronics and Systems (DIMES), University of Calabria, 87036 Rende, Italy
[2] Emerging Nanoscaled Integrated Circuits & Systems (EnICS) Labs, Faculty of Engineering, Bar-Ilan University, Ramat-Gan 5290002, Israel
* Correspondence: esteban.garzon@unical.it

Abstract: Content-addressable memory (CAM) has been part of the memory market for more than five decades. CAM can carry out a single clock cycle lookup based on the content rather than an address. Thanks to this attractive feature, CAM is utilized in memory systems where a high-speed content lookup technique is required. However, typical CAM applications only support exact matching, as opposed to approximate matching, where a certain Hamming distance (several mismatching characters between a query pattern and the dataset stored in CAM) needs to be tolerated. Recent interest in approximate search has led to the development of new CAM-based alternatives, accelerating the processing of large data workloads in the realm of big data, genomics, and other data-intensive applications. In this review, we provide an overview of approximate CAM and describe its current and potential applications that would benefit from approximate search computing.

Keywords: CAM; content-addressable memory; ternary CAM; associative memory; associative processor; approximate CAM

Citation: Garzón, E.; Yavits, L.; Teman, A.; Lanuzza, M. Approximate Content-Addressable Memories: A Review. *Chips* **2023**, *2*, 70–82. https://doi.org/10.3390/chips2020005

Academic Editor: Andrea Boni

Received: 17 January 2023
Revised: 8 March 2023
Accepted: 27 March 2023
Published: 30 March 2023

1. Introduction

Content-addressable memory (CAM) or associative memory is a storage structure that accesses memory by content rather than by location [1]. In addition to the write and read operations that are supported by static random access memory (SRAM) and dynamic RAM (DRAM), CAM allows massively parallel search operations between an input query pattern and the entire dataset stored within the memory architecture. Thanks to this property, CAM is a commonly sought after component for constructing state-of-the-art memory-based systems where high-speed parallel search is required. CAM has been adopted in a wide spectrum of application domains, from network routers and switches to digital signal processing, data analytics, and microprocessors. Within microprocessors, parallel search operations are required by many components, including fully associative cache memory, translation look-aside buffers, branch prediction buffers, and more [1–5]. Conventional CAM is mainly designed to reveal exact matches between the input query pattern and the stored information. However, in the past decade, approximate search has become an attractive feature for different data-intensive applications such as comparison-intensive big data workloads, machine learning, and pattern recognition applications in images, DNA sequencing, and biomedical data [6–10]. For this class of emerging applications, the main aim is to find similar rather than exact matching patterns. In other words, mismatching characters can exist between the query pattern and a stored data entry. Therefore, a certain Hamming distance is tolerable, and such a stored pattern should still be considered a "match". With applications of similarity search growing quickly, approximate search-capable CAM has become a subject of active scientific research. Accordingly, this paper provides a review of approximate search-capable CAM circuit solutions and their main applications.

The target of this manuscript is to familiarize the reader with the state-of-the-art approximate search solutions, present the advantages and drawbacks of past and existing

approximate CAM solutions, discuss their main limitations, and summarize our findings, insights, and conclusions. This paper considers today's data and comparison-intensive applications, state-of-the-art similarity search solutions, and their main design considerations.

This review is organized as follows. Section 2 presents the conventional CAM topologies. Section 3 discusses the different classes of approximate CAM, highlighting their past and current prospects. Section 4 presents some applications of approximate search CAM. Finally, Section 5 summarizes the main conclusions of this work.

2. Background

The following subsections overview the structure and common implementations of conventional CAMs, introducing the challenges and trade-offs that limit their use in approximate search applications. The CAM architecture described below is the basis for any type of approximate CAM implementation. The CAM architecture is used "as is" when implementing software-based approximate solutions. For hardware-based approaches, additional circuitry is considered.

2.1. CAM Hardware Implementation

Figure 1 shows the top-level view of the CAM architecture. It is mainly built from $m \times n$ CAM bit cells (BCs), search data registers (SDRs), and match line sense amplifiers (MLSAs). The CAM cells are based on the structure of the standard six-transistor SRAM bit cells (6T-SRAM), thereby enabling write and read operations through word lines (WLs) and complementary bit lines, which are called search lines (SLs) in the case of CAM structures. Similar to SRAM, both the write and read operations are carried out by asserting the WLs. When a write operation is required, the data are driven onto the SLs through the search data registers or drivers (SDRs). During read access, the data of the CAM cells are sampled from the SLs following the completion of the read by means of SL sensing circuitry (not shown in Figure 1).

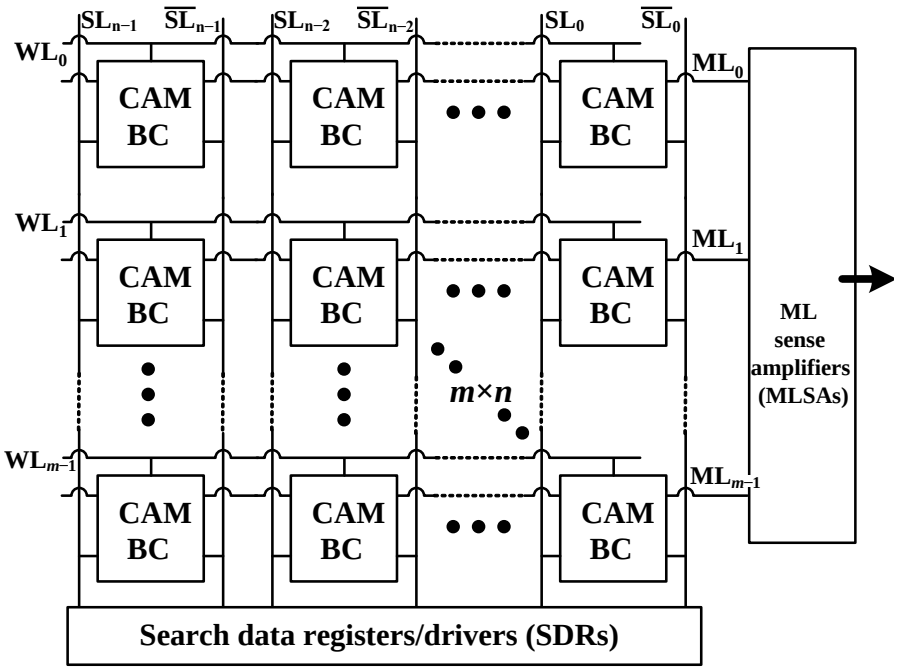

Figure 1. CAM top-level view.

The main difference between CAM and random-access memory is the compare operation. Compare operations are applied by driving the query pattern onto the SLs through the SDRs. A match occurs when the stored data matches the query pattern; otherwise, a mismatch is asserted. Match and mismatch cases depend on the voltage level of the MLs, which are sensed by the MLSAs.

Conventional CAM can be classified as having NOR- or NAND-based architectures. The schematic cells for the NOR and NAND CAM memory structures are shown in Figure 2a,b. For the sake of simplicity, the WL wires are not shown. The core of the cells is the cross-coupled inverter that holds the volatile data within the internal nodes D and $\overline{D}$. The data nodes are connected to XNOR circuitry composed of M_1–M_4 (Figure 2a) and M_1–M_3 (Figure 2b) transistors for the NOR and NAND cells, respectively. For the NOR-type cell, the XNOR operation is carried out by conditionally pulling down the ML through M_1/M_3 (or M_2/M_4). As for the NAND-type cell, the XNOR operation is implemented by conditionally enabling the pass transistor (M_1) through the M_2/M_3 transistors. Figure 2c,d shows the connectivity schemes of CAM memory words based on the NOR ML and NAND ML, along with their MLSA and pre-charge or evaluation circuitry.

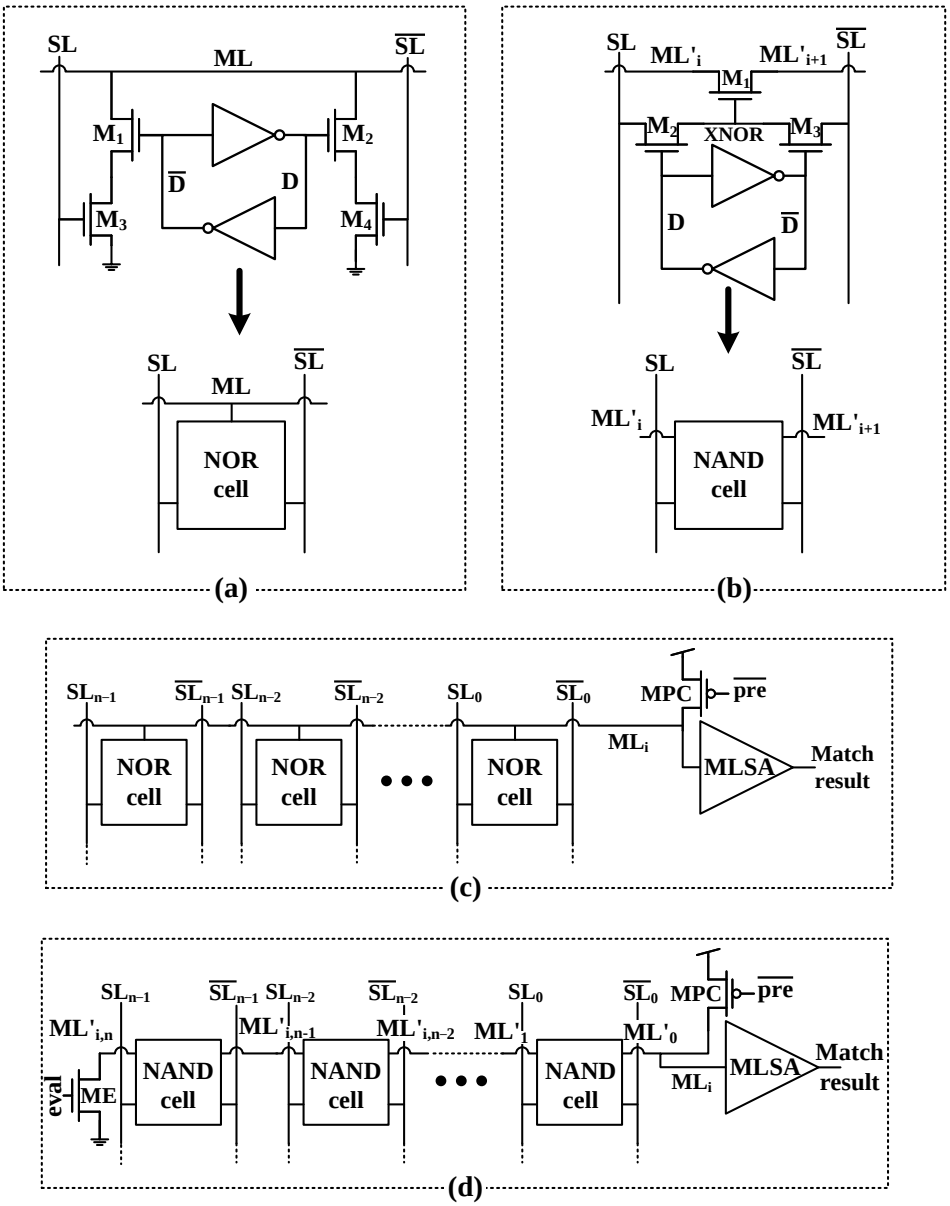

Figure 2. CAM cells based on (**a**) NOR and (**b**) NAND. For the sake of simplicity, word lines (WLs) are not shown in (**a**,**b**). CAM memory words based on (**c**) NOR match line and (**d**) NAND match line (adopted from [1]). The CAM words employ a match line sense amplifier (MLSA).

2.2. CAM Operation Principle

NOR-based and NAND-based CAM carry out the compare operation in two phases: **(1)** pre-charge and **(2)** evaluation. For the NOR-based CAM, the comparison operation is performed through the pre-charge transistor (MPC), while for the NAND-based CAM, the comparison depends on the MPC and ME transistors. To avoid unwanted operations, the SLs have to be discharged down to ground prior to the compare operation. During the pre-charge phase, the $\overline{\text{pre}}$ signal is asserted, and the ML is charged to V_{DD} in both configurations. For a NAND-based CAM, the gate of the ME is driven to ground during precharge. During the evaluation phase, the NOR and NAND CAM behave in a complementary manner.

In the context of an NOR CAM, a search is performed by applying an inverted pattern to the search lines. If any of the stored bits do not match the inverted search bit, then a discharge will occur on one of the match lines (either M_1/M_3 or M_2/M_4), indicating a mismatch. The value of the inverse match line (ML_i) at the end of the comparison cycle will be zero in this case. If the stored bit in every cell is not equal to the inverted search bit, the ML has no path to discharge through. Therefore, the ML voltage level remains high at the end of the comparison cycle, indicating a match. In the case of the NAND CAM, if the stored bit in every cell is equal to the search bit, then the ML is able to discharge through the series of $M_{1,i}$ transistors (refer to Figure 2b), resulting in a drop to zero and a match being signaled at the end of the comparison cycle. If there is even a single mismatch among the NAND CAM cells, then the ML will stay high, indicating that a match has not been found.

2.3. CAM Limitations for Approximate Search

The exact match CAM circuits, which were described above, feature several design challenges that have limited their use for approximate search CAM. These limitations, including accuracy, complexity, and cost, are elaborated upon hereafter:

- **Accuracy:** Conventional CAM memory is built to carry out exact match (rather than approximate match), and thus approximate search is not expressly supported.
- **Complexity:** Implementing approximate match capabilities in conventional CAM memory can be challenging, and it may increase the overall design complexity.
- **Cost:** To add approximate matching capabilities to conventional CAM memory, modifications to the hardware or software may be necessary. This may incur an additional cost.

Overall, while it is possible to carry out approximate search with conventional CAM memory without modifying the memory array structure, dedicated approximate match CAM may be more effective and efficient for this purpose. This is discussed further in the following section.

3. Approximate Content-Addressable Memory

In recent years, there have been numerous proposals for ternary and binary CAM cell designs that utilize NOR and NAND cells. These designs range from CMOS-based approaches to novel memory-based solutions, and most of them are mainly built to support exact match [10–32]. Due to the presence of possible bit errors or "do not care" bits in the data pattern, there are several applications which greatly benefit from approximate search rather than exact search. These applications will be introduced in Section 4.

In approximate search, a certain number of mismatching characters between a query pattern and the data stored in CAM is allowed. If the difference between the stored pattern and the query pattern is below a predetermined threshold, then the comparison should be deemed a "match". In simpler cases, the difference is limited to replacements, meaning that a certain data element is replaced by another. However, in text and sequenced DNA data processing, there are two possible additional variations: insertions and deletions. The former is the insertion of a data element into a sequence of data elements, while the latter

is the deletion of a data element from a sequence. Collectively, these changes to a data sequence (replacements, insertions, and deletions) are known as edits.

While most solutions for approximate search have been focused on software solutions, there have been a considerable number of studies that have proposed hardware-level alternatives.

3.1. Classes of Approximate CAM

In this subsection, we elaborate on the past and current prospects of approximate CAM reported in the state of the art. We discuss various hardware- and software-level solutions proposed for approximate search in CAM. Additionally, we will highlight the limitations and challenges of these solutions and their potential for future improvements. Approximate search CAM classes are represented in Figure 3 and are discussed below.

Figure 3. Classes of approximate CAM.

3.1.1. Error-Tolerant CAM

CAM designs that provide soft error tolerance using error-correcting codes were proposed in [33–35]. The error-tolerant CAM of Pagiamtzis et al. [33] is based on an error-correcting match circuit that utilizes parity bits for each CAM word and modifies the match line-sensing circuitry to tolerate up to one mismatching bit. Krishnan et al. [34] proposeed error-correcting codes for ternary CAM (TCAM) by replacing the match line sense amplifier with an analog comparator. Efthymiou [35] exploited NAND-type CAM along with parity bits and a dedicated ML scheme to tolerate soft errors. These designs mainly target single event upsets, use memory redundancy, and typically tolerate a limited Hamming distance of 1–4 bits. This is at the expense of an increased area footprint.

3.1.2. Minimum Hamming Distance Search CAM

Minimum Hamming distance search CAM is a CAM design that is capable of searching for patterns with a Hamming distance below a predetermined threshold. Mattausch et al. [36] proposed a CAM-based architecture for minimum Hamming distance search. This CAM design considers both digital and analog circuitry for bit and word comparison, followed by winner-take-all (WTA) circuitry. There have also been several proposals for Hamming distance approximation using emerging memory (memristor crossbar) designs. For example,

Zhu et al. [37] and Taha et al. [8] presented a hybrid CMOS/memristor-based CAM design for Hamming network circuits. These architectures mainly exploit a crossbar array and WTA circuitry, allowing the network to identify the pattern or patterns within the dataset that are similar to the query pattern. Other examples of minimum Hamming distance search CAM include NCAM [38] and PPAC [39]. NCAM utilizes near-memory logic to determine the sum of the squares of the differences between data words, while PPAC calculates the Hamming similarity through a population count, which involves tallying the number of ones in the XNOR outputs of the bit cells of the CAM word.

3.1.3. Tunable Sample Time CAM

A tunable sample time CAM utilizes the timing of the score signal delay or the speed of the match line discharge as a measure of the Hamming distance. In [40], whenever a bit mismatch occurred between the bits of the search and query patterns, a delay was added. Therefore, the Hamming distance between the two patterns can be determined by the overall delay of the score signal. Rahimi et al. [41] proposed memristive-based approximate CAM for energy-efficient GPUs. The architecture was experimentally evaluated for image processing kernels, showing a Hamming distance tolerance of up to two bits. Although this was shown to be a cost-effective solution, enabling approximate CAM is accomplished through meticulous timing of the match line discharge. Imani et al. [42] exploited the delay lines at the clock inputs of four sense amplifiers on each match line, allowing a Hamming distance tolerance of up to four bits. These tunable sampling time methods demand precise device and circuit sizing, require almost perfect skew balancing between all ML timing circuits, are sensitive to jitter, and are prone to producing false negatives and false positives. Therefore, the overall accuracy of the approximate search technique is limited.

3.1.4. Locality-Sensitive Hashing

This class of approximate search CAM utilizes near-neighbor algorithms, such as local-sensitivity hashing (LSH) of query patterns and stored data. Ni et al. [43] demonstrated that ferroelectric-based TCAMs can be used to compute the Hamming distance between query and data patterns within the memory itself. Similarly, Riazi et al. [44] and Sheybani et al. [45] exploited LSH-based CAM to enable approximate search in hardware security applications. These designs provide a large Hamming distance tolerance as well as tolerance to edit distances. However, before the storage and search operations, these schemes require local data hashing, and a large Hamming distance does not always result in low similarity for the hashed data sketches [46]. Therefore, LSH limits the precision in approximate search CAM.

3.1.5. Tunable Match Line Discharge Rate CAM

Another class of approximate search CAM utilizes the ML discharge rate. Garzón et al. [47] exploited the conventional NOR-type CAM cell to construct a Hamming distance-tolerant CAM (HD-CAM), which can achieve either exact or approximate matching. The Hamming distance threshold is determined by the combination of the voltage that controls the speed of the ML discharge and the sense amplifier reference voltage. Although HD-CAM is able to tolerate large Hamming distances, it does not have a built-in capability to tolerate the edit distance. To deal with this, Hanhan et al. [48] proposed a novel edit distance-tolerant CAM (EDAM) for approximate search applications. Thanks to its dedicated hardware, EDAM is highly efficient in applications such as text processing and genome analysis. These beneficial features come at the cost of an increased area footprint due to the additional circuitry included within their bit cells.

3.2. Approximate CAM Limitations

Several approximate CAM designs have been proposed in recent decades, as introduced above. Despite their promising capabilities to accelerate comparison-intensive applications, there is still the need for more affordable, cost-effective, and energy-efficient

similarity search-capable solutions. Some limitations (in terms of degree of similarity, speed, complexity, and cost) to using approximate CAM are as follows:

- **Degree of Similarity:** Unlike exact match CAMs, approximate match CAMs aim to retrieve data that are similar but not necessarily identical to the query pattern. Therefore, the focus is on the relevance of the results rather than their exactness. However, this can be a limitation in situations where the required degree of similarity is critical, such as in certain text or image recognition applications. In such cases, it is important to carefully choose the similarity metric and threshold to ensure that the results are relevant enough for the target application.
- **Speed:** While approximate match CAM is generally slower than exact match CAM, the degree of difference in speed can vary depending on several factors. For example, determining the closest match in approximate match CAM requires comparing the query pattern to multiple pieces of data, which can add to the search time. Additionally, some approximate CAM designs, such as those based on a tunable sample time or tunable ML discharge rate, may introduce an additional search delay that affects the overall speed. This is because additional circuitry may be introduced in the match line discharge path [47,48].
- **Complexity and Cost:** Implementation of approximate match CAM can be more complex than exact match CAM, as it requires the use of algorithms or additional circuitry to determine the similarity between the query and search pattern. In fact, to enable approximate match, CAM cells often need to be tailored to a specific application. For example, Garzón et al. [47] and Hanhan et al. [48] used an additional transistor to control the match line discharge path. Therefore, the overall bit cell area footprint increases.

3.3. Power Consumption

CAM and approximate CAM have been shown to have higher power densities than traditional memory arrays, which can limit their usage for certain applications that require low power consumption or have power constraints. For example, in mobile or Internet of Things (IoT) devices that are powered by batteries, power consumption is a critical concern. In fact, CAM is characterized by frequent and simultaneous power-hungry accessing of all memory cells in the array. The power consumption of CAM can be affected by various factors, such as the number of cells, the typology (NOR-type or NAND-type), the search algorithm, the data pattern, and the adopted sensing circuitry. Moreover, the increase in the number of cells (used for larger CAM) can result in significant power dissipation. This issue may limit the use of CAM in power-limited or battery-operated systems.

In terms of its footprint, CAM can be more area-hungry than other memory arrays because it usually requires more transistors per cell than traditional memory types. This makes it a less attractive option for applications where the memory density is constrained, such as in portable devices.

Speed is another important consideration when it comes to CAM. In general, the larger the CAM memory word, the higher the power consumption can be per operation.

Scalability is also an important consideration when it comes to CAM. As the number of cells in a CAM increases, the overall power consumption can increase. This can limit the scalability of CAM for applications that require large databases or energy-efficient search operations.

Overall, advancements in hardware (e.g., energy-efficient match line schemes) and the use of emerging non-volatile memory technologies can make CAM more feasible for power-constrained applications by also improving the scalability of CAM [49].

4. Approximate Search Applications

During the last decade, due to the increasing interest in emerging memory technologies, emerging applications, and the need to accelerate comparison-intensive tasks, approximate search CAM has shown its potential for applications that benefit from approx-

imate match rather than exact match. Some approximate CAM applications are shown in Figure 4, including machine learning, deep learning, data analytics, and computational biology [6–10,50]. From the computational biology realm, genomics has been a subject of great interest, mainly due to the exponential growth of the ever-increasing sequenced data volume [51]. Genome analysis has experienced astonishing growth over the last decade [52], and it is the basis for different kinds of applications, such as monitoring environmental ecosystems and genomics surveillance (e.g., as a tool to fight the COVID-19 pandemic), sustainable agriculture, environment monitoring of Earth, and personalized healthcare [53–56].

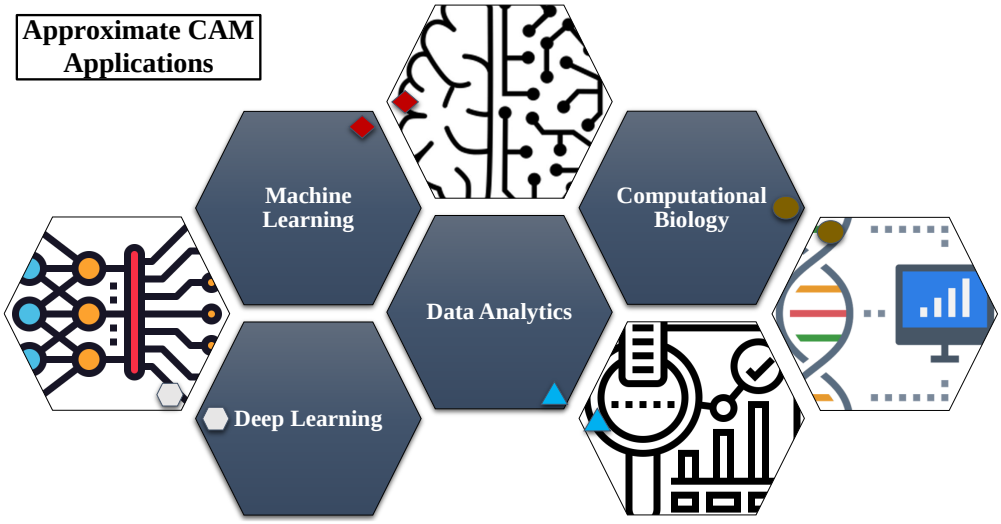

Figure 4. Approximate CAM applications: machine learning, deep learning, data analytics, and computational biology.

4.1. Future Prospects and Challenges for Approximate CAM

Despite the growing interest in approximate CAM, it still faces several challenges that need to be addressed in order to fully implement its potential. One challenge, if needed by the target application, is improving the degree of similarity between the query pattern and the stored data, as this can greatly affect the accuracy of the search results. Another challenge is reducing power consumption, as approximate CAM tends to have a higher power density compared with traditional memory arrays.

Furthermore, approximate CAM needs to be designed and optimized based on the specific requirements and characteristics of the application. For example, in bioinformatics, approximate CAM needs to consider the variability of DNA sequences and account for various types of mutations. Hanhan et al. proposed EDAM, a CMOS edit distance-tolerant content-addressable memory for approximate search [48]. (This is presented as a case study in the next subsection.) The EDAM design presents challenges in scaling to support large genome databases due to its large size (42 transistors) and the required cross-connectivity among neighboring memory columns, which may adversely affect the density and timing.

In summary, while approximate CAM holds promise in enabling error-tolerant search operation in various applications, it still faces challenges related to degree of similarity, power consumption, and design optimization. Future research in this area should focus on addressing these challenges and exploring new energy-efficient approaches for approximate search using CAM.

4.2. Case Study: Genomics and DNA Pattern Detection Using Approximate CAM

Recent advancements in DNA sequencing technology, in conjunction with PCR molecular techniques, have been exploited to detect and characterize the viral DNA strands associated with the current COVID-19 outbreak [57,58]. The procedure entails the process of sequencing a genome, specifically the DNA, which is subsequently utilized in detection

methodologies, such as the polymerase chain reaction (PCR). The DNA of organisms is composed of four nucleotides: adenine (A), guanine (G), cytosine (C), and thymine (T), which are commonly referred to as DNA bases. DNA sequencing is a process of determining the bases of a DNA chain. Current high-throughput DNA sequencing devices possess the capability to simultaneously sequence multiple DNA samples in parallel [59]. The DNA sequencing process and genomic analysis are executed through a series of steps, including [60] (1) preparation of the DNA samples, (2) DNA sequencing, which results in the production of multiple DNA fragments also known as DNA reads, and (3) classification of DNA reads, alignment of DNA reads, genome assembly, and analysis of genetic variants, among others. State-of-the-art tools, such as Kraken and Kraken2 [61,62], are used to classify unknown DNA. However, Kraken operation is based on exact matching of the k-mers in the sequenced DNA patterns (reads) against a DNA database it creates. Thus, to operate with sufficient sensitivity, it requires relatively high coverage (high percentage of the target DNA in a sample), which is not always available in specific reads. For example, DNA reads of viruses tend to include the host's DNA, which significantly impedes the ability to identify the virus's DNA in the sample.

A fast and highly sensitive approximate matching-based DNA classification scheme, known as EDAM, was proposed in [48], and its high-level architecture is shown in Figure 5. The EDAM memory scheme was built with commercial process and based on 6T-SRAM bit cells with additional comparison logic that enables the edit distance tolerance and approximate search capabilities. A comprehensive design space evaluation showed that EDAM is capable of tolerating a range of edit distances, and it was shown to classify more reads than Kraken2 during several experiments on raw reads while also retaining high precision. The EDAM architecture allows for efficient genomic surveillance through its ability to rapidly identify and categorize viral and other pathogen DNA.

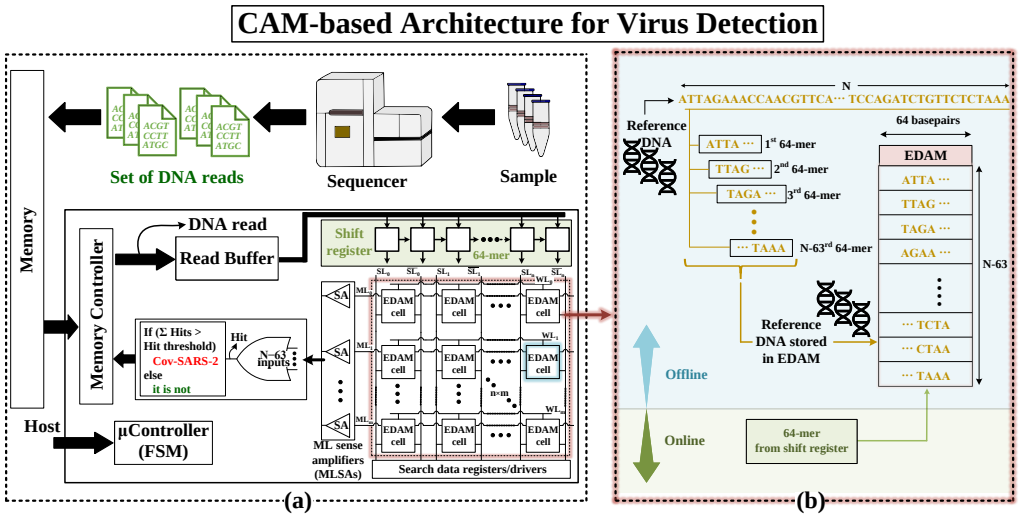

Figure 5. Approximate CAM-based architecture for virus detection and classification [48]. (**a**) EDAM platform. (**b**) The offline construction of the reference DNA database along with the online virus detection operation.

5. Conclusions

This paper presented an overview of approximate content-addressable memory, starting with the background of conventional CAM and then introducing the current approximate CAM classes. Several state-of-the-art alternatives that enable approximate search were also introduced, followed by an overview of the limitations of such approximate CAM methods. We further introduced approximate search as an alternative to accelerate the processing of large data workloads in the realm of big data analytics, computational biology, machine learning, and deep learning applications. Finally, approximate search

CAM was presented through a case study in the genomics application domain, showing its potential for hardware acceleration of genomic surveillance.

Author Contributions: Conceptualization, E.G., L.Y., A.T. and M.L.; investigation, E.G., L.Y., A.T. and M.L.; writing—review and editing, E.G., L.Y., A.T. and M.L. All authors have read and agreed to the published version of the manuscript.

Funding: European Union's Horizon Europe programme for research and innovation under grant agreement No. 101047160; Israeli Ministry of Science and Technology under Lise Meitner grant for Israeli-Swedish research collaboration; Italian Ministry of University and Research (MUR): PRIN 2020LWPKH7.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CAM	Content-addressable memory
SRAM	Static random-access memory
DRAM	Dynamic random-access memory
BC	Bit cell
SDR	Serach data register
ML	Match line
MLSA	Match line sense amplifier
WL	Word line
SL	Search line
D	Data
MPC	Pre-charge transistor
TCAM	Ternary content-addressable memory
WTA	Winner-take-all
LSH	Local sensitivity hashing
HD-CAM	Hamming distance-tolerant CAM
EDAM	Edit distance-tolerant CAM
PCR	Polymerase chain reaction
A	Adenine
G	Guanine
C	Cytosine
T	Thymine

References

1. Pagiamtzis, K.; Sheikholeslami, A. Content-addressable memory (CAM) circuits and architectures: A tutorial and survey. *IEEE J. Solid-State Circuits* **2006**, *41*, 712–727. [CrossRef]
2. Ge, Q.; Yarom, Y.; Cock, D.; Heiser, G. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *J. Cryptogr. Eng.* **2018**, *8*, 1–27. [CrossRef]
3. Basu, A.; Gandhi, J.; Chang, J.; Hill, M.D.; Swift, M.M. Efficient virtual memory for big memory servers. *ACM SIGARCH Comput. Archit. News* **2013**, *41*, 237–248. [CrossRef]
4. Gracioli, G.; Alhammad, A.; Mancuso, R.; Fröhlich, A.A.; Pellizzoni, R. A survey on cache management mechanisms for real-time embedded systems. *ACM Comput. Surv. (CSUR)* **2015**, *48*, 1–36. [CrossRef]
5. Karam, R.; Puri, R.; Ghosh, S.; Bhunia, S. Emerging Trends in Design and Applications of Memory-Based Computing and Content-Addressable Memories. *Proc. IEEE* **2015**, *103*, 1311–1330. [CrossRef]
6. Imani, M.; Kim, Y.; Rahimi, A.; Rosing, T. ACAM: Approximate Computing Based on Adaptive Associative Memory with Online Learning. In Proceedings of the 2016 International Symposium on Low Power Electronics and Design, ISLPED'16, New York, NY, USA, 8–10 August 2016; pp. 162–167. [CrossRef]

7. Ali, M.; Agrawal, A.; Roy, K. RAMANN: In-SRAM Differentiable Memory Computations for Memory-Augmented Neural Networks. In Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design, ISLPED'20, New York, NY, USA, 10–12 August 2020; pp. 61–66. [CrossRef]
8. Taha, M.M.; Teuscher, C. Approximate memristive in-memory Hamming distance circuit. *ACM J. Emerg. Technol. Comput. Syst. (JETC)* **2020**, *16*, 1–14. [CrossRef]
9. Kaplan, R.; Yavits, L.; Ginosasr, R. BioSEAL: In-Memory Biological Sequence Alignment Accelerator for Large-Scale Genomic Data. In Proceedings of the 13th ACM International Systems and Storage Conference, Haifa, Israel, 2–4 June 2020; pp. 36–48. [CrossRef]
10. Kaplan, R.; Yavits, L.; Ginosar, R.; Weiser, U. A Resistive CAM Processing-in-Storage Architecture for DNA Sequence Alignment. *IEEE Micro* **2017**, *37*, 20–28. [CrossRef]
11. Do, A.T.; Chen, S.; Kong, Z.H.; Yeo, K.S. A High Speed Low Power CAM With a Parity Bit and Power-Gated ML Sensing. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2013**, *21*, 151–156. [CrossRef]
12. Mishra, S.; Mahendra, T.V.; Dandapat, A. A 9-T 833-MHz 1.72-fJ/Bit/Search Quasi-Static Ternary Fully Associative Cache Tag With Selective Matchline Evaluation for Wire Speed Applications. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2016**, *63*, 1910–1920. [CrossRef]
13. Arsovski, I.; Patil, A.; Houle, R.M.; Fragano, M.T.; Rodriguez, R.; Kim, R.; Butler, V. 1.4Gsearch/s 2-Mb/mm2 TCAM Using Two-Phase-Pre-Charge ML Sensing and Power-Grid Pre-Conditioning to Reduce Ldi/dt Power-Supply Noise by 50%. *IEEE J. Solid-State Circuits* **2018**, *53*, 155–163. [CrossRef]
14. Yavits, L.; Morad, A.; Ginosar, R. Computer Architecture with Associative Processor Replacing Last-Level Cache and SIMD Accelerator. *IEEE Trans. Comput.* **2015**, *64*, 368–381. [CrossRef]
15. Dong, Q.; Jeloka, S.; Saligane, M.; Kim, Y.; Kawaminami, M.; Harada, A.; Miyoshi, S.; Yasuda, M.; Blaauw, D.; Sylvester, D. A 4 + 2T SRAM for Searching and In-Memory Computing with 0.3-V V_{DDmin}. *IEEE J. Solid-State Circuits* **2018**, *53*, 1006–1015. [CrossRef]
16. Chan, Y.S.; Huang, P.T.; Wu, S.L.; Lung, S.C.; Wang, W.C.; Hwang, W.; Chuang, C.T. 0.4 V Reconfigurable Near-Threshold TCAM in 28 nm High-k Metal-Gate CMOS Process. In Proceedings of the 2018 31st IEEE International System-on-Chip Conference (SOCC), Arlington, VA, USA, 4–7 September 2018; pp. 272–277. [CrossRef]
17. Chang, Y.J.; Liao, Y.H. Hybrid-Type CAM Design for Both Power and Performance Efficiency. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2008**, *16*, 965–974. [CrossRef]
18. Sethi, D.; Kaur, M.; Singh, G. Design and performance analysis of a CNFET-based TCAM cell with dual-chirality selection. *J. Comput. Electron.* **2017**, *16*, 106–114. [CrossRef]
19. Cheng, K.H.; Wei, C.H.; Chen, Y.W. Design of low-power content-addressable memory cell. In Proceedings of the 2003 46th Midwest Symposium on Circuits and Systems, Cairo, Egypt, 27–30 December 2003; Volume 3, pp. 1447–1450. [CrossRef]
20. Do, A.T.; Yin, C.; Yeo, K.S.; Kim, T.T.H. Design of a power-efficient CAM using automated background checking scheme for small match line swing. In Proceedings of the 2013 Proceedings of the ESSCIRC (ESSCIRC), Bucharest, Romania, 16–20 September 2013; pp. 209–212. [CrossRef]
21. Agarwal, A.; Hsu, S.; Mathew, S.; Anders, M.; Kaul, H.; Sheikh, F.; Krishnamurthy, R. A 128 × 128 b high-speed wide-and match-line content addressable memory in 32 nm CMOS. In Proceedings of the 2011 Proceedings of the ESSCIRC (ESSCIRC), Helsinki, Finland, 12–16 September 2011; pp. 83–86. [CrossRef]
22. Jothi, D.; Sivakumar, R. Design and analysis of power efficient binary content addressable memory (PEBCAM) core cells. *Circuits, Syst. Signal Process.* **2018**, *37*, 1422–1451. [CrossRef]
23. Garzón, E.; Yavits, L.; Finocchio, G.; Carpentieri, M.; Teman, A.; Lanuzza, M. A Low-Energy DMTJ-based Ternary Content-Addressable Memory with Reliable Sub-Nanosecond Search Operation. *IEEE Access* **2023**, *11*, 16812–16819. [CrossRef]
24. Hussain, S.W.; Mahendra, T.V.; Mishra, S.; Dandapat, A. Match-Line Division and Control to Reduce Power Dissipation in Content Addressable Memory. *IEEE Trans. Consum. Electron.* **2018**, *64*, 301–309. [CrossRef]
25. Prasanth, K.; Ramireddy, M.; Ravindrakumar, S. High speed, low matchline voltage swing and search line activity TCAM cell array design in 14 nm FinFET technology. In *Emerging Trends in Electrical, Communications, and Information Technologies*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 465–473. ._38. [CrossRef]
26. Zackriya, V.M.; Kittur, H.M. Precharge-Free, Low-Power Content-Addressable Memory. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2016**, *24*, 2614–2621. [CrossRef]
27. Ramanathan, A.K.; Rangachar, S.S.; Hung, J.M.; Lee, C.Y.; Xue, C.X.; Huang, S.P.; Hsueh, F.K.; Shen, C.H.; Shieh, J.M.; Yeh, W.K.; et al. Monolithic 3D+-IC Based Massively Parallel Compute-in-Memory Macro for Accelerating Database and Machine Learning Primitives. In Proceedings of the 2020 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 12–18 December 2020; pp. 28.5.1–28.5.4. [CrossRef]
28. Yavits, L.; Kaplan, R.; Ginosar, R. GIRAF: General Purpose In-Storage Resistive Associative Framework. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *33*, 276–287. [CrossRef]
29. Kaplan, R.; Yavits, L.; Ginosar, R. PRINS: Processing-in-Storage Acceleration of Machine Learning. *IEEE Trans. Nanotechnol.* **2018**, *17*, 889–896. [CrossRef]
30. Yavits, L.; Kvatinsky, S.; Morad, A.; Ginosar, R. Resistive Associative Processor. *IEEE Comput. Archit. Lett.* **2015**, *14*, 148–151. [CrossRef]

31. Yavits, L.; Morad, A.; Ginosar, R. Sparse matrix multiplication on an associative processor. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *26*, 3175–3183. [CrossRef]
32. Garzón, E.; Teman, A.; Lanuzza, M.; Yavits, L. AIDA: Associative In-Memory Deep Learning Accelerator. *IEEE Micro* **2022**, *42*, 67–75. [CrossRef]
33. Pagiamtzis, K.; Azizi, N.; Najm, F.N. A Soft-Error Tolerant Content-Addressable Memory (CAM) Using An Error-Correcting-Match Scheme. In Proceedings of the IEEE Custom Integrated Circuits Conference 2006, San Jose, CA, USA, 10–13 September 2006; pp. 301–304. [CrossRef]
34. Krishnan, S.C.; Panigrahy, R.; Parthasarathy, S. Error-Correcting Codes for Ternary Content Addressable Memories. *IEEE Trans. Comput.* **2009**, *58*, 275–279. [CrossRef]
35. Efthymiou, A. An error tolerant CAM with nand match-line organization. In Proceedings of the 23rd ACM International Conference on Great Lakes Symposium on VLSI, Paris, France, 2–3 May 2013; pp. 257–262. [CrossRef]
36. Mattausch, H.; Gyohten, T.; Soda, Y.; Koide, T. Compact associative-memory architecture with fully parallel search capability for the minimum Hamming distance. *IEEE J. Solid-State Circuits* **2002**, *37*, 218–227. [CrossRef]
37. Zhu, X.; Yang, X.; Wu, C.; Wu, J.; Yi, X. Hamming network circuits based on CMOS/memristor hybrid design. *IEICE Electron. Express* **2013**, *10*, 20130404. [CrossRef]
38. Del Mundo, C.C.; Lee, V.T.; Ceze, L.; Oskin, M. Ncam: Near-data processing for nearest neighbor search. In Proceedings of the 2015 International Symposium on Memory Systems, Washington, DC, USA, 5–8 October 2015; pp. 274–275. [CrossRef]
39. Castañeda, O.; Bobbett, M.; Gallyas-Sanhueza, A.; Studer, C. PPAC: A versatile in-memory accelerator for matrix-vector-product-like operations. In Proceedings of the 2019 IEEE 30th International Conference on Application-specific Systems, Architectures and Processors (ASAP), New York, NY, USA, 15–17 July 2019; Volume 2160, pp. 149–156. [CrossRef]
40. Bui, T.T.; Shibata, T. A Low-Power Associative Processor with the R-th Nearest-Match Hamming-Distance Search Engine Employing Time-Domain Techniques. In Proceedings of the 2010 Fifth IEEE International Symposium on Electronic Design, Test Applications, Ho Chi Minh City, Vietnam, 13–15 January 2010; pp. 54–57. [CrossRef]
41. Rahimi, A.; Ghofrani, A.; Cheng, K.T.; Benini, L.; Gupta, R.K. Approximate associative memristive memory for energy-efficient GPUs. In Proceedings of the 2015 Design, Automation Test in Europe Conference Exhibition (DATE), Grenoble, France, 9–13 March 2015; pp. 1497–1502. [CrossRef]
42. Imani, M.; Rahimi, A.; Kong, D.; Rosing, T.; Rabaey, J.M. Exploring hyperdimensional associative memory. In Proceedings of the 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA), Austin, TX, USA, 4–8 February 2017; pp. 445–456. [CrossRef]
43. Ni, K.; Yin, X.; Laguna, A.F.; Joshi, S.; Dünkel, S.; Trentzsch, M.; Müller, J.; Beyer, S.; Niemier, M.; Hu, X.S.; et al. Ferroelectric ternary content-addressable memory for one-shot learning. *Nat. Electron.* **2019**, *2*, 521–529. [CrossRef]
44. Riazi, M.S.; Samragh, M.; Koushanfar, F. Camsure: Secure content-addressable memory for approximate search. *ACM Trans. Embed. Comput. Syst. (TECS)* **2017**, *16*, 1–20. [CrossRef]
45. Sheybani, N.; Zhang, X.; Hussain, S.U.; Koushanfar, F. SenseHash: Computing on Sensor Values Mystified at the Origin. *IEEE Trans. Emerg. Top. Comput.* **2022**. . [CrossRef]
46. Marçais, G.; DeBlasio, D.; Pandey, P.; Kingsford, C. Locality-sensitive hashing for the edit distance. *Bioinformatics* **2019**, *35*, i127–i135. [CrossRef]
47. Garzón, E.; Golman, R.; Jahshan, Z.; Hanhan, R.; Vinshtok-Melnik, N.; Lanuzza, M.; Teman, A.; Yavits, L. Hamming Distance Tolerant Content-Addressable Memory (HD-CAM) for DNA Classification. *IEEE Access* **2022**, *10*, 28080–28093. [CrossRef]
48. Hanhan, R.; Garzón, E.; Jahshan, Z.; Teman, A.; Lanuzza, M.; Yavits, L. EDAM: Edit distance tolerant approximate matching content addressable memory. In Proceedings of the 49th Annual International Symposium on Computer Architecture, New York, NY, USA, 18–22 June 2022; pp. 495–507. [CrossRef]
49. Garzón, E.; Lanuzza, M.; Teman, A.; Yavits, L. AM^4: MRAM Crossbar Based CAM/TCAM/ACAM/AP for In-Memory Computing. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2023**, *13*, 408–421. [CrossRef]
50. Li, W.; Ye, X.; Wang, D.; Zhang, H.; Tang, Z.; Fan, D.; Sun, N. PIM-WEAVER: A High Energy-efficient, General-purpose Acceleration Architecture for String Operations in Big Data Processing. *Sustain. Comput. Inform. Syst.* **2019**, *21*, 129–142. [CrossRef]
51. Stephens, Z.D.; Lee, S.Y.; Faghri, F.; Campbell, R.H.; Zhai, C.; Efron, M.J.; Iyer, R.; Schatz, M.C.; Sinha, S.; Robinson, G.E. Big data: Astronomical or genomical? *PLoS Biol.* **2015**, *13*, e1002195. [CrossRef]
52. Khatamifard, S.K.; Chowdhury, Z.; Pande, N.; Razaviyayn, M.; Kim, C.; Karpuzcu, U.R. Read Mapping Near Non-Volatile Memory. *arXiv* **2017**. [CrossRef]
53. Glasl, B.; Bourne, D.G.; Frade, P.R.; Thomas, T.; Schaffelke, B.; Webster, N.S. Microbial indicators of environmental perturbations in coral reef ecosystems. *Microbiome* **2019**, *7*, 94. [CrossRef]
54. Singh, B.K.; Trivedi, P.; Egidi, E.; Macdonald, C.A.; Delgado-Baquerizo, M. Crop microbiome and sustainable agriculture. *Nat. Rev. Microbiol.* **2020**, *18*, 601–602. [CrossRef]
55. Zhang, Q.; Difford, G.; Sahana, G.; Løvendahl, P.; Lassen, J.; Lund, M.S.; Guldbrandtsen, B.; Janss, L. Bayesian modeling reveals host genetics associated with rumen microbiota jointly influence methane emission in dairy cows. *ISME J.* **2020**, *14*, 2019–2033. [CrossRef]

56. Alser, M.; Bingöl, Z.; Cali, D.S.; Kim, J.; Ghose, S.; Alkan, C.; Mutlu, O. Accelerating genome analysis: A primer on an ongoing journey. *IEEE Micro* **2020**, *40*, 65–75. [CrossRef]
57. Bloom, J.S.; Sathe, L.; Munugala, C.; Jones, E.M.; Gasperini, M.; Lubock, N.B.; Yarza, F.; Thompson, E.M.; Kovary, K.M.; Park, J.; et al. Swab-Seq: A high-throughput platform for massively scaled up SARS-CoV-2 testing. *medRxiv* **2020.**
58. Artika, I.M.; Wiyatno, A.; Ma'roef, C.N. Pathogenic viruses: Molecular detection and characterization. *Infect. Genet. Evol.* **2020**, *81*, 104215. [CrossRef] [PubMed]
59. Illumina. Illumina—DNA Sequencing. 2021. Available online: https://www.illumina.com/ (accessed on 15 December 2022)
60. Kim, J.S.; Cali, D.S.; Xin, H.; Lee, D.; Ghose, S.; Alser, M.; Hassan, H.; Ergin, O.; Alkan, C.; Mutlu, O. GRIM-Filter: Fast seed location filtering in DNA read mapping using processing-in-memory technologies. *BMC Genom.* **2018**, *19*, 89. [CrossRef]
61. Wood, D.E.; Salzberg, S.L. Kraken: Ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.* **2014**, *15*, R46. [CrossRef] [PubMed]
62. Wood, D.E.; Lu, J.; Langmead, B. Improved metagenomic analysis with Kraken 2. *Genome Biol.* **2019**, *20*, 257. [CrossRef] [PubMed]