sensors

MDPI

*Review*

# A Comprehensive Review of Hardware Acceleration Techniques and Convolutional Neural Networks for EEG Signals

**Yu Xie [1] and Stefan Oniga [1,2,*]**

[1] Faculty of Informatics, University of Debrecen, 4032 Debrecen, Hungary; yu.xie@inf.unideb.hu
[2] North University Center of Baia Mare, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania
* Correspondence: oniga.istvan@inf.unideb.hu

**Abstract:** This paper comprehensively reviews hardware acceleration techniques and the deployment of convolutional neural networks (CNNs) for analyzing electroencephalogram (EEG) signals across various application areas, including emotion classification, motor imagery, epilepsy detection, and sleep monitoring. Previous reviews on EEG have mainly focused on software solutions. However, these reviews often overlook key challenges associated with hardware implementation, such as scenarios that require a small size, low power, high security, and high accuracy. This paper discusses the challenges and opportunities of hardware acceleration for wearable EEG devices by focusing on these aspects. Specifically, this review classifies EEG signal features into five groups and discusses hardware implementation solutions for each category in detail, providing insights into the most suitable hardware acceleration strategies for various application scenarios. In addition, it explores the complexity of efficient CNN architectures for EEG signals, including techniques such as pruning, quantization, tensor decomposition, knowledge distillation, and neural architecture search. To the best of our knowledge, this is the first systematic review that combines CNN hardware solutions with EEG signal processing. By providing a comprehensive analysis of current challenges and a roadmap for future research, this paper provides a new perspective on the ongoing development of hardware-accelerated EEG systems.

**Keywords:** EEG signal analysis; convolutional neural networks; feature extraction; hardware acceleration



**Citation:** Xie, Y.; Oniga, S. A Comprehensive Review of Hardware Acceleration Techniques and Convolutional Neural Networks for EEG Signals. *Sensors* **2024**, *24*, 5813. https://doi.org/10.3390/s24175813

Academic Editor: Boon Giin Lee

Received: 23 July 2024
Revised: 3 September 2024
Accepted: 4 September 2024
Published: 7 September 2024



## 1. Introduction

Electroencephalography (EEG) has become a fundamental area of neuroscience research, clinical diagnostics, and brain–computer interface (BCI) applications. EEG signals can record the activity of the brain's electrical waves and are a key area of interest for practitioners studying neural activity in the human brain and the BCI. Applications of EEG include emotion classification, motor imagery, epilepsy detection, sleep monitoring, attention monitoring, cognitive load assessment, BCI control systems, neurofeedback training, detection of pathological brain activity, and diagnosis of mental disorders. These applications include but are not limited to medical diagnosis, neurorehabilitation, mental health monitoring, education and learning assessment, augmented and virtual reality interaction, user status monitoring, and many other fields. As the relevance of EEG extends across various domains, there is a growing demand for proficient and efficient signal-processing techniques to fully harness the latent capabilities of EEG data. Previous reviews related to EEG have mostly focused on software development. For example, these reviews covered general deep learning methods based on EEG [1], analyzed transfer learning mechanisms for common EEG tasks [1], and listed existing software solutions [2]. However, beyond these software-focused discussions, it is also crucial to consider application scenarios that demand small size, low power consumption, high security, and high accuracy, such as motor imagery and emotion detection. The authors of [3] briefly discussed hardware solutions for EEG-based emotion recognition and introduced classifiers like support vector machines

(SVMs), k-nearest neighbor (KNN), and artificial neural networks (ANNs), providing a performance comparison. Unfortunately, they did not extensively cover the hardware design of deep learning methods. Despite some similarities, our work has the following distinct features:

(1) This paper focuses on the challenges of hardware acceleration in EEG-based systems, which we believe is a critical area in the development of wearable medical solutions.
(2) We conduct a systematic analysis of EEG signal characteristics and categorize existing feature extraction methods, which helps identify the most suitable hardware acceleration strategies for different application scenarios and provides a clear direction for further research.
(3) To the best of our knowledge, this paper is the first to systematically review the combination of CNN hardware solutions with EEG signal processing, filling a gap in this specialized field.
(4) This paper comprehensively considers the development of EEG and CNN technologies, identifies existing challenges, and provides insights into future development directions, offering a new perspective for research in this area.

### *1.1. EEG Signal-Processing Challenges*

The characteristics of EEG signals include high temporal resolution and a direct reflection of neural activity [4]. However, they present unique challenges due to their low spatial resolution, sensitivity to noise, and high dimensionality. Extracting meaningful information from EEG data necessitates complex signal-processing techniques. EEG signal processing generally includes preprocessing, parameter determination, and feature extraction. There is currently no established set of paradigms for EEG signal processing.

Non-deep learning methods commonly used in EEG signal processing include techniques such as SVM, kNN, and Linear Discriminant Analysis (LDA), as well as traditional feature extraction approaches like Independent Component Analysis (ICA) and Principal Component Analysis (PCA). While these non-deep learning methods are indeed simpler, faster, and often easier to implement on wearable devices, they may struggle to achieve high accuracy due to the inherent complexity and continuous nature of EEG signals. These traditional methods typically rely on handcrafted features, the selection of which is highly dependent on the researcher's expertise and can be time-consuming. Additionally, these models often have limited analytical power, which may not be sufficient for handling complex tasks. In contrast, deep learning greatly simplifies the processing workflow by automatically extracting features while simultaneously performing decoding. Although deep learning models do involve a large number of parameters, resulting in significant computational demands, this is precisely where hardware acceleration becomes particularly valuable. Hardware acceleration enables the efficient execution of these models, making it feasible to deploy deep learning approaches in real-time EEG signal-processing applications, even in resource-constrained environments. Recent advances in deep learning have demonstrated remarkable capabilities in various fields, such as image processing and natural language understanding [5]. These advancements have spurred interest in utilizing deep learning for EEG signal analysis. Convolutional neural networks (CNNs) are one of the most common applications of deep learning, particularly in image processing. Images, as a form of feature representation, allow for the visualization of EEG signals' temporal and frequency domain characteristics through additional spatial information. This enables CNNs to fully exploit the characteristics of EEG data contained in the input. In addition, CNNs can autonomously learn discriminative features from EEG data, potentially reducing reliance on manual feature engineering and enhancing classification performance. It is, therefore, natural for EEG researchers to attempt to replicate this approach.

### *1.2. Hardware Acceleration and Efficiency*

While CNNs offer promising solutions for EEG signal analysis, they come with substantial computational demands, particularly in real-time applications like BCIs. One

possible approach is to deploy processing tasks to cloud servers with more powerful resources [6,7]. But in many cases, applications such as real-time emotion monitoring and motor imagery need to respond immediately to changes in the user's state. Relying on cloud servers introduces network latency. If there is network fluctuation or no network, the real-time requirements may not be met. In addition, EEG data usually contain sensitive information, and transmitting these data to the cloud for processing may raise privacy and security concerns. Therefore, for many wearable EEG devices, hardware acceleration is necessary for achieving key functions such as low power consumption, real-time processing, data security, and independent operation. These factors make hardware acceleration particularly important for wearable devices. Hardware acceleration, especially on Field-Programmable Gate Arrays (FPGAs) and Graphics Processing Units (GPUs), can significantly enhance the speed and efficiency of EEG processing based on CNNs [8]. FPGAs, in particular, provide a reconfigurable hardware platform that allows researchers to optimize CNN architectures for specific EEG applications. FPGA-based implementations are renowned for their parallel processing capabilities, low latency, and energy efficiency. These characteristics make FPGAs an excellent choice for real-time EEG processing [9].

### *1.3. Research Objectives and Structure*

The primary objective of this paper is to provide a comprehensive review of the integration of CNNs and hardware acceleration techniques for EEG signal processing. To achieve this goal, we have structured this paper into several sections. Section 2 reviews EEG-based hardware solutions, including preprocessing, feature extraction, and various features used in EEG analysis. Section 3 discusses efficient CNN architectures, including pruning, quantization, tensor decomposition, knowledge distillation, and neural architecture search. Section 4 explores the challenges in this field and provides recommendations for future research directions. Section 5 summarizes the essential findings and insights of this paper. The purpose of this review is to explore the latest achievements in the field of EEG and CNNs and to help more researchers gain a deeper understanding of the hardware solutions, potentials, and challenges in this field.

## 2. EEG-Based Hardware Solutions

EEG signals are variable and contain a lot of noise. As researchers study EEG, they have developed many algorithms to extract raw signals and convert them into usable information. This process comprises several critical stages, each employing a unique set of techniques and technologies. In this section, we present the state-of-the-art hardware implementations of various algorithms used in the preprocessing and feature extraction stages.

### *2.1. Preprocessing*

Before exploring the wealth of information hidden within EEG signals, it is typically necessary to preprocess the data to address the substantial noise and artifacts that often accompany them [10,11]. While some datasets offer EEG data with certain artifacts already removed, the hardware implementation of preprocessing blocks provides a more comprehensive and practical evaluation of the hardware resources required for EEG-based solutions.

These artifacts can be both physiological and non-physiological. Physiological artifacts are associated with activities such as head movements (EMG), blinking (EOG), muscle contractions, heart activity (ECG), sweating, and respiration (RS) [12,13]. Non-physiological artifacts occur due to external factors like electrode malfunctions (Ep), power sources (AC), ventilation, magnetic fields, cable movements (Cable), and more [14,15]. As summarized in Table 1, the spectrum of EEG signals spans from 0 to 100 Hz and intertwines with various artifacts across this range.

The frequency range of EEG bands is defined slightly differently in different studies. It is worth noting that in Table 1, the frequency bands of Mu waves and Alpha waves overlap because the Mu band is a special form of the Alpha band. Alpha waves are among the most easily recognizable in EEG and are usually found in the posterior region of the head (occipital lobe) when the subject is relaxed or has their eyes closed. Mu waves usually appear in the sensory-motor area of the human brain. When the subject is at rest, Mu waves can be clearly observed; however, when the subject is moving or exercising, the amplitude of the Mu wave decreases or disappears, which is called motor inhibition. Although the Alpha and Mu bands are very similar, they come from different brain regions and have distinct associated functions. The authors of [16] showed that ICA can help distinguish activities from different brain regions, thereby differentiating Alpha waves from Mu waves.

**Table 1.** Precision and recall scores for the proposed model.

| EEG Band | Frequency (Hz) |
|---|---|
| Alpha | 8–13 |
| Beta | 12–30 |
| Delta | 0.5–3.5 |
| Theta | 4–7 |
| Mu | 8–12 |
| Gamma | 26–100 |
| **Artifacts** | **Frequency (Hz)** |
| ECG | 0.5–40 |
| EOG | 0.1–17 |
| EMG | 0.1–2.5/20–100 |
| RS | 0.1–2.5 |
| Ep | 0.1–35 |
| Cable | 2–15 |
| AC | 50 |

One method for reducing noise in EEG signals is the application of digital filters. In [17], a bandpass filter ranging from 8 to 45 Hz was used. While this configuration successfully eliminated some low-frequency artifacts, it came at the cost of attenuating EEG activity in the Theta and Delta brainwaves. In contrast, [18,19] employed a low cutoff frequency of 2 Hz. This lenient setting allowed for the coexistence of slow eye-movement artifacts with the EEG recordings. Ref. [20] used an 81-order digital filter, which provided a better graphical representation for the analysis of Beta and Mu brainwaves and was successfully implemented on the DE0-Nano-Soc development card. It is worth noting that using only bandpass filtering may not cover complex interference scenarios caused by actual artifacts. Another method was described in [21], which involves three key steps: downsampling, bandpass filtering, and integrating a Mean Average Reference (MAR) filtering block. Downsampling, performed at twice the rate without altering the frequency content, reduces the power spectral amplitude by half and enhances signal clarity. The bandpass filter operates within the range of 2 Hz to 60 Hz, ensuring that EEG signals remain within the desired spectrum. The MAR filter involves re-referencing electrode signals relative to their collective average, effectively reducing the signal's dependence on specific electrode locations [4]. This is achieved by subtracting the average signal value from each electrode signal. With this technique, artifacts associated with the physical reference electrode are effectively removed, while averaging acts as noise suppression, preventing the introduction of new artifacts into the EEG signal.

On the other hand, in [22], the authors initiated an adaptation strategy employing ICA based on kurtosis. This technique can extract artifacts co-occurring within the same frequency band as EEG signals. However, the application of this method is entirely offline, and the feasibility of its hardware implementation remains unconfirmed. In [23], fast ICA was employed for epileptic seizure detection from multi-channel signals. This rapid

ICA assists in artifact removal and reduces energy dissipation. Eigenvalue decomposition (EVD), used in conjunction with the Jacobi algorithm, resulted in a 77.2% reduction in area. The practical performance of this architecture was implemented on an FPGA and validated through appropriate databases. These blind source separation algorithms have become popular methods for hardware implementations of EEG preprocessing [24,25].

Nevertheless, the efficacy of these methods is related to the number of independent components obtained, a quantity dependent on the number of signal sources or electrodes. Consequently, these methods exhibit limitations in situations with a small number of sources. This realization has prompted the exploration of cost-effective alternatives [26,27] for integrating the ICA model, where the authors developed a performance evaluation method that fused matched filtering and threshold cross-checking based on the number of electrodes.

In the field of EEG-based classification, effective methods must encompass a set of techniques to ensure signal quality, including but not limited to detrending, re-referencing, outlier detection, interpolation, detection and correction, and elimination of artifacts.

### *2.2. Feature Extraction*

Preprocessed EEG data include a wealth of information. However, to use this information, it must first be extracted and converted into meaningful features. This section reviews feature extraction techniques that transform EEG signals into quantifiable attributes. We examine five types of feature extraction used in this domain: frequency-based features, time-based features, time-frequency features, spatial features, and raw signals.

#### 2.2.1. Frequency-Based Features

Frequency-based features analyze the rhythmic patterns of brain activity. Given that a large number of studies have identified correlations between spectral components and brainwave rhythms, researchers have a clear tendency to focus on frequency characteristics [5,28].

The complexity of related algorithms varies greatly across software platforms, and researchers rarely consider the issue of computing resource consumption at the software level. Consequently, their high complexity levels make many of them unsuitable for direct hardware implementation. However, methods such as Power Spectral Density (PSD), Fourier Transform, Discrete Cosine Transform (DCT), Singular Spectrum Analysis (SSA), and combinations of filter banks still hold promise for integration into hardware solutions, as the hardware design of these traditional signal-processing modules is well established. Hardware-accelerated Fast Fourier Transform (FFT) modules, with their moderate resource requirements, can efficiently compute PSD. In contrast, higher-order spectral (HOS) analysis, Linear-Frequency Cepstral Coefficients (LFCCs), Mel-Frequency Cepstral Coefficients (MFCCs), the Burg method, and Auto-Regressive Reflection Coefficients (ARRCs), while effective, might require more resource allocation and design work for successful integration into hardware solutions.

Cross-frequency coupling (CFC) functions are promising approaches that leverage synchronized variations across different frequency bands [29,30]. These functions manifest as connectivity graphs or indices. They not only ensure robustness but also have the potential to reduce EEG data rates and memory requirements. However, the computational complexity of CFC algorithms and the hardware demands, such as Coordinate Rotation Digital Computer (CORDIC) processors for computing cross-frequency Hilbert transforms, pose significant challenges for ultra-low-power hardware designs [31].

It is foreseeable that PSD has become one of the most popular methods in the EEG classification domain. In [18,21], PSD approximations were used, with only one power value per channel calculated based on the average activity within the beta band (12 to 30 Hz). While this approach does not necessitate FFT accelerators, making it more hardware-friendly, it introduces information loss within the frequency bands. In contrast, [22] employed an FFT

with 128 frequency bins, ensuring sufficient resolution to capture fluctuations both within and between the EEG wavebands.

#### 2.2.2. Time-Based Features

In addition to frequency-based features, time-based features are also a key research direction in EEG signal processing because they can reflect the dynamic changes in brain activity over time and provide indicators, such as maximum and average amplitudes, slopes, and skewness, that reflect transient brain activity.

Notably, Higher-Order Crossings (HOCs) have become a common approach. HOCs are based on tracking zero crossings and their consecutive differences, offering valuable insights into the temporal dynamics of EEG signals. Higuchi's Fractal Dimension (HFD) is well known for its ability to capture the non-linear aspects of data, making it particularly suitable for feature extraction in various EEG events. Other time-based feature extraction methods include Hjorth parameters, Grassberger–Procaccia (GP), Hybrid Adaptive Filtering (HAF), the Hurst exponent, Local Binary Patterns (LBPs), and Event-Related Late Positive Potentials (LPPs).

Compared to frequency-domain approaches, the advantage of time-based methods is their relatively lower hardware resource requirements, making them an attractive choice for hardware implementations. However, it must be acknowledged that methods involving non-linear transformations, such as HFD or Hjorth parameters, often increase the complexity of hardware design. In [19], a time-based approach was employed for feature extraction. The method focuses on extracting HOCs and skewness (SK) information from time series to construct feature vectors. To facilitate hardware-friendly implementation, [19] introduced the Approximate SK Indicator (ASKI), significantly reducing gate counts by a factor of 86. For HOCs, the authors employed a straightforward implementation based on comparators, using 3-bit SR triggers to monitor the previous values of a given sample. Identifying sign changes between consecutive sample values triggers a simple counter to accumulate zero crossings. Both of these metrics are standardized and fed into a dense neural network. Hjorth parameters indicate statistical characteristics in EEG signal analysis. Calculating them requires variance computation for the "activity" parameter, the square root of the ratio for the "mobility" parameter, and division operations for the "complexity" parameter with first-order derivatives [32].

Sample Entropy (SE) is another time-based method [33] that has proven to be relatively hardware-friendly due to its simplicity. SE is a measure of time-series complexity defined as the logarithm of a ratio, making its hardware implementation relatively straightforward. Ref. [34] proposed that SE architecture runs ten times faster on FPGAs than on CPUs. Methods with minimal hardware resource utilization and power consumption include statistical mean and other time-based methods like Root Mean Square (RMS), Variance (Var), and Standard Deviation (SD), as observed in the articles we surveyed, both on software and hardware platforms, these methods all have an average accuracy of no more than 60%. Therefore, they are the least ideal options.

#### 2.2.3. Time-Frequency Features

The fusion of time-domain and frequency-domain techniques gives rise to time-frequency features, which are adept at handling the non-stationary nature of EEG data. Wavelet Transforms (WTs) and their various forms (such as Discrete Wavelet Transform (DWT), Continuous Wavelet Transform (CWT), and Adaptive Tunable Q-Wavelet Transform) play a significant role. The Short-Time Fourier Transform (STFT) and the Hilbert–Huang Transform (HHT), along with its Independent Mode Function (IMF) and Empirical Mode Decomposition (EMD), have also garnered attention. Unlike Fourier and Wavelet Transforms, the HHT employs a unique approach using EMD to decompose signals, extracting quasi-harmonics starting from the highest frequency. This method demonstrates robustness in analyzing non-stationary and non-linear time series, a common characteristic of EEG signals [35]. However, it is worth noting that the calculation of the IMF in the HHT

involves an iterative algorithm. The parameters of each layer of this algorithm are highly correlated, making it difficult to take advantage of hardware parallel processing, especially in some real-time processing application scenarios.

Wavelet Transforms (WTs) and fundamental forms of the STFT are two feasible choices for hardware implementation. Ref. [36] proposed an STFT design based on the Xilinx Zedboard and extracted 20 frequency feature components for classification. These 20 features were divided into five groups, each corresponding to five different EEG wave frequency bands. Features were extracted from 128 channels of EEG signals at a clock rate of 50 MHz. Ref. [37] employed the STFT for time-frequency analysis of EEG segments and extracted the regions of interest in terms of frequency bands and features, implementing a deep learning model and an STFT module on an FPGA to enhance seizure detection. Ref. [38] introduced a generalized spectrogram method. The STFT was applied to signals captured from six electrodes, which were subsequently mixed based on asymmetric indices. Ref. [39] implemented a DWT with a two-class LDA classification on the Basys 3 Artix-7 FPGA Board. Ref. [40] presented a design and implementation of the Morlet CWT for EEG signals on the Spartan 3AN FPGA. By optimizing the trade-off between speed and silicon area, it could generate all-scale wavelet coefficients for a 1024-point EEG signal in approximately one millisecond when designed to run at a maximum clock speed of 125 MHz. In the STFT, the signal to be processed is divided into multiple shorter signals according to the size of the window, and a separate FFT is then applied to each small signal. In contrast, the WT processes the complete signal. Both methods yield similar results, with the Wavelet Transform being particularly effective in handling non-periodic and fast-changing features. In contrast, the STFT is favored for real-time processing due to its shorter processing time windows. Moreover, advances in FFT hardware acceleration and various space-efficient techniques for compressed FFT accelerators make them highly attractive for hardware-based solutions.

#### 2.2.4. Spatial Features

Unlike independent features, spatial features depend on the availability of time-based, frequency-based, or time-frequency EEG data for computation. This section introduces a well-known spatial feature method: Common Spatial Pattern (CSP). In a two-classification problem, CSP aims to find spatial filters that maximize the variance of EEG signals for one class while minimizing it for the other class. This results in spatial filters that enhance discriminative features for the two classes, making classification more effective. Here, a simplified explanation of the formula application of CSP is presented. The spatial filter w is calculated by simultaneous diagonalization of the sample covariance matrices from both classes, as follows:

$$J(w) = \frac{w^T \overline{C_1} w}{w^T \overline{C_2} w} \tag{1}$$

where $T$ represents the transpose. $\overline{C_1}$ and $\overline{C_2}$ denote the average covariance matrices of the two different conditions or classes, respectively, and are defined as follows:

$$\overline{C_k} = \frac{1}{N} \sum_{n=1}^{N_k} \frac{D_{(k,n)} D_{(k,n)}^T}{Trace\left(D_{(k,n)} D_{(k,n)}^T\right)},\ k = 1,\ 2 \tag{2}$$

where $Trace\left(D_{(k,n)} D_{(k,n)}^T\right)$ represents the solution of the matrix trace. $N_k$ is the number of samples for the $k$th class, that is, the number of single-trial data. $D_{(k,n)}$ represents the $n$th trial data of the $k$th class. The CSP algorithm frames this as a generalized eigenvalue problem:

$$\overline{C_2}^{-1} \overline{C_1} w = \lambda w \tag{3}$$

The conventional CSP feature extraction method derives features by computing the logarithm of variances from spatially filtered signals, as follows:

$$f_p = \log\left\{\frac{var(Z_p)}{\sum_{i=1}^{2m} var(Z_i)}\right\},\ p = 1,\ 2,\ \cdots,\ 2m \tag{4}$$

where the first and last $m$ rows of the EEG signals are usually taken as the final spatial filter. Here, $Z$ is given by $Z = w^T D$.

In [41], the CSP model was implemented on an Altera Stratix-IV FPGA and verified on the BCI competition dataset. In [42], CSP was used as the feature extraction algorithm with Mahalanobis Distance (MD) as the classifier, and the entire signal-processing task was executed on software embedded in a Nios II processor within a Stratix IV FPGA. This approach reduced FPGA resource consumption but introduced a time delay of 0.5 s. An improvement on the approach in [41] can be found in [42], where the authors accelerated some functions in hardware but still performed most processing on the Nios II, reducing the delay to 0.399 s. In [43], the authors proposed a complete hardware implementation of a BCI embedded on an FPGA using Scalp-Coupled Common Spatial Spectral Patterns (SCSSPs) as the feature extraction algorithm and support vector machines (SVMs) as the classifier.

The previous section mentioned the use of asymmetric indices as a subset of spatial features. These indices are used to compare the relationship between the two hemispheres and thus indicate asymmetric neural activity [44]. This method depends on the calculation of asymmetric indices, which are usually computed between electrode pairs and can be effectively used alongside frequency, time, or time-frequency features. Hardware designs, such as those in [18,21], combine asymmetric indices with channel-based PSD averages, a concept discussed earlier in the frequency-based feature category. In these implementations, utilizing eight channels, the asymmetric indices are calculated based on the PSD averages of electrode pairs.

In addition to CSP and asymmetric indices, another subset of spatial-based features (specific to EEG-based emotion detection systems implemented in software) includes connectivity features, particularly Directed Transfer Function (DTF) features. DTF features serve as causal measurements used to determine brain connectivity patterns [45]. The DTF quantifies the causal influence of one EEG channel on another at specific frequencies. While theoretically, the DTF can be implemented in hardware, it presents more complex challenges compared to asymmetric indices. This complexity arises from the need to calculate the transfer matrices of multivariate autoregressive models and to perform square root and division operations.

#### 2.2.5. Raw Signals

This section introduces some works on EEG classification using raw signals. Due to the complexity of EEG signals, it is difficult for traditional EEG classifiers to obtain good results from raw signals. The shift in researchers' views is mainly due to the popularity of deep neural network (DNN) classifiers. In these new DNN approaches, raw EEG signals can be directly input into the network for feature extraction and classification. Currently, there are no hardware-based implementations that follow this paradigm. This is mainly due to the need to compensate for the lack of specific features in the complexity of DNNs [46,47]. Moreover, most DNN hardware implementations for EEG tend to adopt shallow DNNs to achieve designs that save more computing resources and space. These networks lack the direct capability to process raw EEG data. However, some new material technologies may change this status, such as near/super-threshold techniques in Fully Depleted Silicon on Insulator (FDSOI) technology [48] or adaptive body biasing in advanced semiconductor technologies [49]. These technologies have the potential to facilitate hardware implementations of DNNs that can directly operate on raw EEG signals in wearable classifiers.

Another approach is to use a DNN only as a feature extractor instead of as a classifier. An example can be found in [50], where overlapping filters, as well as Differential Entropy (DE) and PSD analysis, were employed to derive suitable kernels that guide a CNN in automatically extracting features from raw EEG data. The advantage of this approach lies in implicitly merging spectral and entropy information when dealing with raw EEG data for learning and classification.

Table 2 shows the methods discussed in this section.

**Table 2.** Feature extraction methods.

| Feature Domain | Authors | Algorithm | Work Description |
|---|---|---|---|
| Frequency-based | [18,21] | PSD | A power spectrum approximation per channel<br>Hardware friendly, no FFT accelerator required<br>Easily loses band information |
| | [22] | PSD | Dataset: Own FEXD<br>Hardware: Digilent Atlys with Spartan-6<br>An FFT accelerator with 128 frequency bins<br>Captures fluctuations within and between frequency bands |
| | [31] | CFC | Uses connection diagrams to represent EEG<br>Reduces the need to store and compute EEG<br>CORDIC required Hilbert transform of each frequency band |
| Time-based | [19] | HOC, SK | Hardware: TSMC 0.18 um 1P6M CMOS chip<br>Databases: DEAP and SEED<br>Uses ASKI to reduce gate counts, 3-bit SR triggers |
| | [32] | Hjorth | Hardware: Zynq-7000 FPGA<br>Dataset: Bonn University EEG database<br>Classifier: KNN |
| | [33,34] | SE | For statistics, complexity, and irregularity systems<br>For analyzing short and noisy datasets<br>Relatively hardware-friendly |
| Time-frequency | [36] | STFT | Hardware: Xilinx Zedboard<br>Extracts 20 frequency-band features from 128 channels |
| | [37] | STFT | Used for epilepsy detection<br>Only simulation; not applied to real FPGA chips |
| | [38] | STFT | Hardware: ADVANTEST V93000 PS1600 chip<br>Dataset: DEAP<br>STFT features mixed with asymmetric indices |
| | [39] | DWT | Hardware: Basys 3 Artix-7 FPGA Board<br>Classifier: LDA<br>Suitable for developing low-cost, marketable products |
| | [40] | CWT | Hardware: Spartan 3AN FPGA<br>Efficient architecture in Fourier space via optimized speed and silicon area |

**Table 2.** *Cont.*

| Feature Domain | Authors | Algorithm | Work Description |
|---|---|---|---|
| Spatial | [41] | CSP | Hardware: Altera FPGA platform (Stratix-IV)<br>Dataset: motor imagery tasks in a BCI competition<br>Classifier: MD<br>Time delay of 0.5 s |
| | [42] | CSP | Improvement on method in [41]; reduces the delay to 0.399 s |
| | [43] | SCSSP | Hardware: Virtex-6 FPGA<br>Dataset: BCI Competition IV-dataset 2a<br>Classifier: SVM |
| | [18] | Ai, PSD | Based on a scaled version of IHPR with customized LUT |
| | [21] | Ai, PSD | Improvement on method in [18]; reduces the gate and power consumption |
| | [45] | DTF | Describes incidental effect between two channels<br>Difficult to implement in hardware |
| Raw | [50] | PSDCNN | Dataset: Own<br>For emotion recognition and fatigue-driving |

"Ai" denotes asymmetric indices.

## 3. Efficient Convolutional Neural Networks

In Section 2, we mentioned that DNNs have been widely used in the study of EEG signals. CNNs have showcased remarkable capabilities in various domains, from image recognition to natural language understanding. Their innate ability to autonomously learn hierarchical features has positioned them as the predominant force in software-driven DNNs. The application of CNNs presents an opportunity to fundamentally reshape the approach to neural data analysis, interpretation, and classification. It is well recognized that the surprising progress of CNNs has been accompanied by a large number of parameters that need to be calculated. In particular, the real-time calculation of CNNs is challenging in the field of BCIs. This section introduces five highly efficient CNN hardware techniques specifically used for EEG signal processing: pruning, quantization, tensor decomposition, knowledge distillation, and neural architecture search. The applications we reviewed are summarized at the end of this section.

### *3.1. Pruning*

One approach to reducing the computational complexity of CNNs is pruning, a technique dating back to the early 1990s [51]. Researchers have found that there are some redundant or less informative neural nodes in CNNs. Therefore, deleting these unimportant weights, filters, channels, and even layers during the inference process has little effect on accuracy. The core idea of pruning is to identify and eliminate redundancy in the network to reduce computational and memory requirements. This process can be categorized into two main strategies: weight pruning and structural pruning.

#### 3.1.1. Weight Pruning

Weight pruning is a core technique in the design of efficient neural networks. Removing some unimportant connections does not affect the performance of the network because not all neural nodes are equally important. Several strategies have been developed to effectively execute weight pruning.

Early work in this field, such as Optimal Brain Damage (OBD) [51] and Optimal Brain Surgeon (OBS) [52], introduced the concept of utilizing second-order derivatives (Hessian matrix) of the loss function to identify unnecessary weights. These methods laid the foundation for subsequent weight-pruning techniques.

A widely adopted three-step deep learning method was proposed in [53,54]. First, the network is initially trained to identify essential connections. Subsequently, less important connections are pruned. Finally, the network undergoes retraining to fine-tune the remaining connections. This iterative process gradually leads to a more compressed network structure. Furthermore, dead neurons, with all their connected weight pruned, can be safely discarded. Some research has suggested that pruned larger models can outperform smaller dense models in terms of accuracy under the same memory constraints [55,56]. While traditional weight pruning methods operate in the spatial domain, [57] proposed a frequency-domain approach that transforms spatial weight into frequency-domain coefficients. Pruning is performed dynamically in different frequency bands during each iteration, leading to higher compression ratios.

In the previously mentioned methods, a global magnitude threshold is applied to the entire DNN, which can affect network performance since applying the same compression rate to each layer is not always optimal. Some studies have used variable pruning rates to target layers of different importance. Ref. [51] combined selective learning, identifying weight importance concerning the loss, and discarding other weights by cutting off the gradient flow. Ref. [58] employed a Gaussian Mixture Model (GMM) for each layer's weight distribution to determine layer-wise compression rates. Pruning layers are selected based on the number of weights, with small amplitudes estimated using the GMM.

Additionally, amplitude-aware pruning methods like [59] introduce considerations for energy reduction along with compression ratios and accuracy loss when determining pruning strategies.

Weight-pruning techniques remove less important weights or neurons, regardless of their position within the network. This process results in so-called "unstructured" networks. Such networks may require specialized software or hardware adaptations, thus limiting the universality of the weight-pruning algorithm on hardware.

#### 3.1.2. Structural Pruning

On the other hand, structural pruning focuses on eliminating entire filters, channels, or even whole layers. This approach not only reduces computational requirements but also leads to more compact network architectures, which is especially valuable in resource-constrained environments. Unlike weight pruning, structural pruning does not require specialized hardware or software support. Because it only changes the size of the input and output, it does not change the parameters within the layer. This method maintains strong compatibility. Mainstream structural pruning strategies primarily aim to identify and remove unimportant filters. They are typically categorized into three main branches:

- Criterion-Based Pruning: In this method, filters are ranked based on specific pruning criteria. Methods that consider both L1 and L2 norm criteria, such as L1-norm-based pruning [60] and Soft Filter Pruning (SFP) [61], have been explored. Additionally, activation neuron criteria, such as the Average Percentage of Zeros (APoZ) [62], help identify and remove filters with low APoZ values.
- Ranking-Based Pruning: HRank [63] employs feature map ranking as the metric for filter pruning. Lower-ranked feature maps are considered to contain less information, making them one of the primary candidates for pruning.
- Optimization-Based Pruning: Methods like ThiNet [64] and NISP [65] formulate pruning as an optimization problem. ThiNet prunes filters in the current layer while minimizing the reconstruction error in the next layer. On the other hand, NISP focuses on reducing the reconstruction error in the "Final Response Layer" (FRL).

In addition to filter and channel pruning, structural pruning extends to whole-layer pruning methods [66–68]. These methods selectively remove layers from the network using various criteria, resulting in more compact models. Similarly, these pruned models not only reduce the amount of computation and time but also maintain good accuracy.

The above algorithms all follow the conventional format of training models, pruning, and fine-tuning, except for the method in [69]. In structural pruning, fine-tuning pruned

models may produce performance comparable to or even worse than training the same model from scratch with randomly initialized weights. Innovative techniques like network slimming [70] and group LASSO loss [71] automatically identify and prune unimportant channels or neurons during the training process, effectively achieving model compression without the need for post-pruning fine-tuning.

### 3.2. Quantization

Quantization is another technique aimed at improving the efficiency of CNNs. It reduces the precision of network weights and activations, typically from high-precision formats (e.g., FP32) to lower-bit fixed-point representations. Precision reduction helps lower memory requirements and computational complexity, making CNNs more hardware-friendly. A commonly used CNN hardware implementation strategy for EEG signal processing is to use quantization and pruning together to achieve better computational efficiency.

In the quantization process, neural network data are constrained to a set of discrete levels, offering various distribution options, including uniform or non-uniform distributions. Uniform distribution [72] (characterized by even step sizes) and non-uniform distribution [73] (typically represented by logarithmic distributions) are commonly used quantization schemes. Quantization can be achieved using deterministic methods and stochastic methods. Deterministic methods (such as projecting data onto the nearest discrete levels [71]) ensure precision, while stochastic methods involve probabilistically determining which of the two neighboring discrete levels the data are projected onto.

The roots of quantizing neural network data can be traced back to the early 1990s [74]. This technology offers two primary forms: post-training quantization [75,76] and quantization-aware training [77–79]. In post-training quantization, FP32 weights and activations are converted to lower-precision formats after the model's full training. Conversely, quantization-aware training incorporates the quantization error as part of the training loss, typically resulting in improved model accuracy.

The field of quantization has achieved impressive results. Research has indicated that FP32 precision parameters can be effectively reduced to INT8 without significant loss of accuracy [75]. Some researchers have even developed 4-bit quantization approaches, eliminating the need for fine-tuning post-quantization [76]. Additionally, experiments using INT8 precision for training and inference have shown competitive results, such as a 1.5% accuracy loss in the case of the ResNet-50 model [77]. Furthermore, advancements like generalized bit precision [78] provide greater flexibility by allowing different bit depths for storing weights and activations, offering more flexibility than traditional INT8 quantization. For example, [78] used 1-bit weights and 2-bit activations to quantize AlexNet, achieving a top-1 accuracy of 51%.

Quantization-aware INT8 training methods [79] enhance the practicality and performance of quantization in deep neural networks by cleverly optimizing calculations in both forward and backward passes through the thoughtful incorporation of loss-aware compensation and parameterized range clipping.

Within the scope of quantization strategies, binarization represents the most extreme form, reducing data to only two possible values: (0, 1) or (−1, 1). This radical reduction allows for resource-efficient XNOR and bit-count operations instead of resource-intensive matrix multiplications. Therefore, binarization is particularly suitable for deploying deep neural networks on resource-constrained devices. However, this extreme quantization inevitably results in significant information loss. Additionally, the discontinuous nature of binary values presents optimization challenges for binary neural networks.

In recent years, to address these challenges, researchers have developed several promising algorithms. Innovative methods like BinaryConnect [80], binary neural networks (BNNs) [81], and XNOR-Net [82] have garnered attention in the field of binary neural networks. BinaryConnect introduces random techniques for binarizing neural networks, applying binarization during both forward and backward propagation. BNNs extend BinaryConnect by binarizing activations, marking a significant milestone in binary neural

network research. Both BinaryConnect and BNNs employ combinations of deterministic and stochastic binarization functions to simplify hardware implementations. In contrast, XNOR-Net employs a different approach, using scaling factors for binary parameters to approximate floating-point parameters. This allows the weight quantization in XNOR-Net to be represented as $w \approx \alpha * bw$, where $\alpha$ represents the floating-point scaling factor of the binary weight $bw$.

Recent advancements include optimization-based binarization techniques. Methods like XNOR-Net and DoReFa-Net [72] concentrate on mitigating quantization errors during the training process. For instance, DoReFa-Net quantizes gradients to expedite training. In addition to local layer considerations, other binarization techniques, such as loss-aware binarization [83] and incremental network quantization [84], directly minimize the overall binary weight loss across the entire network. Through the application of an adaptive error decay estimator, IR-Net [85] represents a pioneering solution for preserving information flow during forward and backward propagation.

Different quantization strategies provide researchers with more options, requiring them to strike a balance between computational efficiency and model accuracy based on specific tasks in EEG signal processing.

### *3.3. Tensor Decomposition*

Tensor decomposition is a technique aimed at breaking down weight tensors into smaller tensors or matrices. This process significantly reduces the number of parameters in CNNs while preserving accuracy. DNN parameters within convolutional layer weight tensors and fully connected layer weight matrices typically exhibit low-rank characteristics [86]. Based on this observation, the directions of algorithm design by researchers can be categorized into two main classes: low-rank matrix decomposition and tensor decomposition.

#### 3.3.1. Low-Rank Matrix Decomposition

The core of low-rank matrix decomposition techniques lies in approximating the weight matrices of DNN layers by multiplying several low-rank matrices. Singular Value Decomposition (SVD) [87] is the most popular low-rank approximation method, emphasizing the description of a matrix through its singular values.

SVD decomposes a matrix $A \in R$ into three key components:

$$A = UDV \tag{5}$$

where $U \in R$ and $V \in R$ are orthogonal matrices, and $D \in R$ is a diagonal matrix containing singular values. A compact and efficient network model is achieved by retaining only the most critical components in the decomposition matrices.

In practice, SVD-based techniques have played a crucial role in various applications. For instance, in [88], SVD was utilized to decompose the product of weight matrices and input data, further enhancing model efficiency. The authors of [89] introduced sparsity into low-rank factorized matrices by maintaining a lower rank for less critical neurons, thus improving compression rates. Additionally, in [90], channel-wise SVD decomposition of convolutional layers was employed to enhance efficiency by subdividing kernels into two consecutive layers of different sizes.

Using SVD at every training node results in high computational costs. Therefore, it becomes important to determine the importance of each DNN layer during training. Innovative approaches have emerged to address these issues. In [91], an SVD training technique was proposed to explicitly achieve low-rank approximation without invoking SVD at every training step. Furthermore, [92] introduced a joint matrix factorization scheme that simultaneously decomposes layers with similar structures and optimizes based on SVD.

#### 3.3.2. Tensorized Decomposition

Tensors, multi-way data arrays, provide a universal foundation for representing complex data. In the context of deep learning, these tensors encompass various orders, with two-dimensional matrices representing second-order tensors commonly used in fully connected layers. Meanwhile, the weight tensors of convolutional layers are represented as fourth-order tensors. Higher-rank tensors require higher compression ratios. Tensor decomposition extends the idea of low-rank matrix decomposition to high-order tensors, breaking down weight tensors into smaller tensors and thus reducing the network's complexity. Prominent methods in this field include Tucker decomposition [93], CANDECOMP/PARAFAC (CP) decomposition [94,95], tensor train (TT) decomposition [96–98], and tensor ring (TR) decomposition [99]. In [93], Tucker decomposition was used to compress convolutional weight kernels with predefined ranks. Simultaneously, the CP decomposition strategically breaks down convolutional kernels into a collection of first-order tensors, reducing parameters and training time. This method requires an iterative process that cleverly combines decomposition with fine-tuning for each convolutional layer. Refs. [94,95] differ in whether they decompose all convolutional layers. The former only decomposes a few convolutional layers, while the latter decomposes all convolutional layers. Ref. [100] introduced a robust low-rank decomposition technique that merges Tucker decomposition with CP decomposition. In this approach, the core tensors extracted through Tucker decomposition are further decomposed using CP decomposition. Other tensor decomposition techniques, such as TT decomposition, appear to be more favorable for recurrent neural networks (RNNs) [96], as their application can significantly reduce RNN model parameters by up to 1000 times. However, they face precision–tradeoff issues in CNN compression. Nevertheless, the authors of [97] proposed a TT format suitable for CNN models. Additionally, TT decomposition was found to be practical for compressing three-dimensional CNNs (3DCNNs) in [98], where an innovative approach for selecting TT ranks to maximize compression gains was outlined. Another pioneering approach involves utilizing TR decomposition, as shown in [99], which employs a progressive genetic algorithm for optimizing rank selection.

In summary, CP decomposition represents tensors as the sum of first-order tensors, offering a unique perspective on compression. On the other hand, Tucker decomposition breaks down tensors into groups of matrices and a compact core tensor. TT decomposition constructs complicated three-dimensional tensor structures, particularly adept at handling high-order tensors. TR decomposition is an extension of TT decomposition, emerging as a linear combination of TT decompositions. These tensor decomposition techniques continue to advance the efficiency of EEG signal processing.

### *3.4. Knowledge Distillation*

Knowledge distillation is a technique that transfers knowledge from large, complex CNNs (teachers) to smaller, more efficient CNNs (students). The student network is trained to mimic the behavior of the teacher, resulting in a compact model with comparable performance. This concept can be traced back to the pioneering work in [101] and has since been extended to the context of deep learning [102]. The core challenge of knowledge distillation revolves around the techniques used to transfer knowledge from the teacher model to the student model, which involves three fundamental components—knowledge, distillation algorithms—and the architecture defining the relationship between the teacher and student models. In this context, knowledge manifests in various forms, including logits, activations, or features extracted from intermediate layers of the teacher model. The distillation algorithms can be categorized as offline, online, or self-distillation.

Offline distillation [101,103–105] extracts knowledge from a pre-trained teacher model and uses the soft label output of the teacher model to train the student network. The authors of [103] used data augmentation to exploit the output distributions of multiple teacher networks. The authors of [101] introduced a tailored distillation approach for quantized models, demonstrating that quantized student networks can closely match

the accuracy of full-precision teacher networks while achieving high compression rates and inference acceleration. In contrast, [104] pioneered a data-free technique, training the student network using synthetic data responses from the complex teacher network. Online distillation [106–108] occurs during the simultaneous training of both the teacher and student models. It employs online knowledge distillation using the soft-label outputs of the teacher network. Ref. [106] proposed training the student model at different checkpoints of the teacher model until convergence is achieved. Meanwhile, Collaborative Learning Knowledge Distillation (KDCL) [107] dynamically generates high-quality soft targets through an ensemble approach for one-stage online training.

Furthermore, as observed in [105,108], knowledge distillation extends its influence to the intermediate layers of the teacher network. Layer Selective Learning (LSL) [105] provides a two-layer selection scheme, known as the inter-layer Gram matrix and layered inter-class Gram matrix, enabling the selection of intermediate layers in both the student and teacher networks for knowledge distillation. In [108], an updated ensemble-based knowledge distillation method was used to leverage features from intermediate layers, facilitating the simultaneous training of the distilled student system and the ensemble of teachers without the need for pre-trained teacher models.

On the other hand, an efficient neural network architecture design for self-knowledge distillation without training a large teacher network has been developed [109,110]. The authors of [109] introduced an auxiliary self-learning network that transfers refined knowledge to the classifier network using soft labels and intermediate feature maps. In [110], effective augmentation strategies were employed to extract knowledge using the best-performing student network from past epochs while simultaneously training the current student network, thereby enhancing network generalization.

In knowledge distillation, hardware constraints can be incorporated during the teacher model's training of the student model to guide it toward meeting specific hardware performance goals. The authors of [111] used knowledge distillation to develop NEMOKD. This system balances latency, accuracy, training time, and hardware costs. It was evaluated on the Movidius Myriad X VPU and Xilinx's programmable Z7020 FPGA. This hardware-aware distillation process minimizes memory usage and computational load, making it possible for the student model to run efficiently on wearable EEG devices.

### *3.5. Neural Architecture Search*

Neural Architecture Search (NAS) is an automated approach for discovering the optimal neural network architecture. NAS defines a search space that includes various CNN models and performance evaluation algorithms. It generally does not require human intervention to discover the most efficient CNN model for specific needs. Unlike methods such as pruning, quantization, and tensor decomposition, as discussed above, NAS requires a specific model. Early NAS methods [112] initially generated a multitude of candidate network architectures based on predefined search space criteria. Each architecture undergoes rigorous training until convergence and is then ranked based on its accuracy on a validation dataset. These rankings provide feedback for fine-tuning the search strategy and generating new neural architectures. This iterative process continues until specified termination conditions are met. Ultimately, the most promising architectures are meticulously selected and evaluated using a test dataset. Since the search space contains thousands of neural networks, training and evaluating each network incurs significant computational and time costs.

Researchers need to find ways to identify efficient network architectures and reduce the computational costs required for training and evaluation. Current NAS methods typically involve three primary stages: training a super network, training and evaluating sampled networks, and finally, improving the discovered networks.

#### 3.5.1. Search Space

The complexity increases exponentially when searching for networks and their interconnected structures in the NAS space [112]. Therefore, the search space needs to be limited. A limited number of network structures are used to form iterative and repeated units. Examples of this approach include NASNet [113], ENAS [114], and AutoDispNet [115]. For instance, AutoDispNet focuses on finding two different cell types, namely normal cells and reduction cells, within a predefined overall network framework. In contrast, ENAS involves an approach where all sub-models share weights collectively, significantly increasing GPU processing time but achieving over a thousandfold improvement compared to traditional NAS [112].

Depending on the specific task, we hope to find a balance between latency (inference time) and memory usage while maintaining satisfactory accuracy. MnasNet [116] is a platform-aware NAS tailored for mobile devices. It studies complex network architectures and analyzes the optimal trade-off between latency and accuracy.

#### 3.5.2. Search Algorithm

In the complex field of NAS, search strategies shape the exploration of a specified search space. Typically, NAS involves a generator that generates sample architectures and an evaluator responsible for assessing their post-training performance. Various search algorithms, such as reinforcement learning (RL) and neural evolution algorithms, have emerged to address this essential issue. RL has achieved significant success, with NAS-RL [112] and MetaQnn [117] demonstrating classification accuracy in image classification tasks. Neuroevolution algorithms originally covered the optimization of neural structures and weights [118], but recently, the focus has shifted primarily toward optimizing neural structures [119]. Ref. [119] showed that both reinforcement learning and neural evolution consistently outperform random search (RS) in terms of final test accuracy. Specifically, neuroevolution consistently manages to strike a balance between high performance and compactness.

#### 3.5.3. Performance Evaluation Strategy

The strategy for evaluating neural architecture performance is a fundamental aspect of the NAS field. In the early days of NAS, the approach was to assess the quality of each sampled network through exhaustive training from scratch [112]. However, this process proved to be highly costly in terms of computational resources and time. Recent advancements have introduced innovative techniques to streamline the performance evaluation step while maintaining high-quality results. Progressive NAS [120] and ReNAS [121] have ingeniously addressed this challenge by constructing accuracy predictors. These predictors are trained using data collected from a subset of sampled network candidates within the search space. During the search process, the trained predictor can evaluate the searched network at no additional cost. Another approach is known as one-shot NAS [122]. It reduces the evaluation cost by training a super network. Each sampled network inherits weights from the super network so that no additional training cost is required. This weight-sharing mechanism is at the core of one-shot NAS and plays a crucial role in reducing the computational burden associated with NAS, thereby significantly improving efficiency.

Automatically designed networks are very powerful but are not suitable for deployment on wearable EEG devices. Recognizing this challenge, hardware-aware NAS methods [123,124] have recently emerged. Unlike previous search strategies, they incorporate hardware factors, such as inference time, power consumption, and memory consumption, into the NAS search process. Ref. [125] proposed a two-stage automated hardware-aware search algorithm, which was evaluated under identical latency constraints across CPU, DPU, and VPU platforms. Compared to manually designed search spaces, this approach achieved higher accuracy. Ref. [126] introduced MemNAS, a search method that takes memory requirements into account. By progressively growing and pruning the network, this algorithm maintains a balance between accuracy and memory consumption across devices

with varying resource constraints. To further reduce evaluation costs, hardware-aware NAS focuses on training a single network that can adapt to different hardware architectures, achieving a balance between high performance and cost-effectiveness.

#### 3.5.4. EEG-Based Convolutional Neural Network Hardware Acceleration

In this subsection, we discuss the available hardware architectures suitable for CNNs. Next, we briefly review some cases showing representative CNN accelerators for EEG signal processing. Over the past decade, CNN hardware acceleration technology has developed significantly, mainly due to the emergence of high-performance GPUs, which are able to cope with the increasing memory requirements and computational complexity of CNNs. Although early applications of deep learning mostly focused on large computing platforms, there is now an increasing need to deploy CNNs on edge devices with limited hardware resources and energy, such as portable or wearable EEG signal-processing devices. To this end, hardware solutions for CNNs have expanded from general-purpose architectures (such as CPUs and GPUs [127]) to spatial architectures (such as FPGAs [22,128] and ASICs [129,130]).

The authors of [128] described the implementation of the VGG model on a PYNQ Z1 FPGA development board for motor imagery tasks. Figure 1 shows a CNN design based on a Xilinx FPGA. The authors found that quantizing from a 32-bit floating-point format to a 16-bit fixed-point format reduced BRAM and DSP resource usage by 25%, with no significant loss in accuracy. In contrast, quantizing to an 8-bit fixed-point format resulted in a 7% decrease in accuracy. The authors of [130] implemented the STFT and CNN models on a TSMC 28nm chip and tested them using the DEAP dataset for emotion recognition during movie-watching tasks.

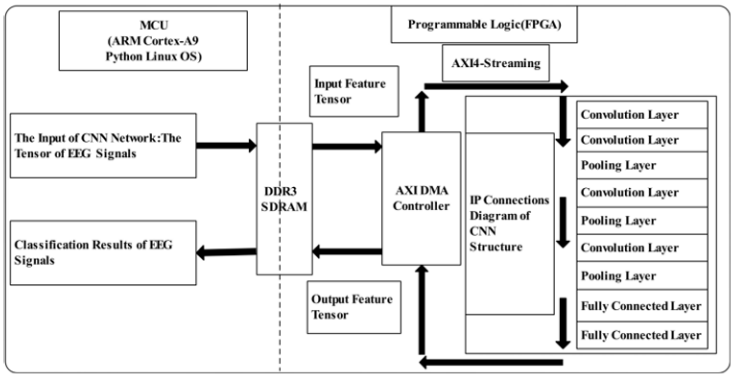


**Figure 1.** FPGA block diagram from [128].

The authors of [129] used a 180 nm 1P6M CMOS as a coprocessor to accelerate the CNN inference. The accelerator achieved 97.8% accuracy in floating-point operations and 93.5% accuracy in fixed-point operations. As shown in Figure 2, the CNN is transferred to the edge device after training. The authors also defined a coprocessor interface (CCI) to facilitate communication between the RISC-V core and the accelerator.

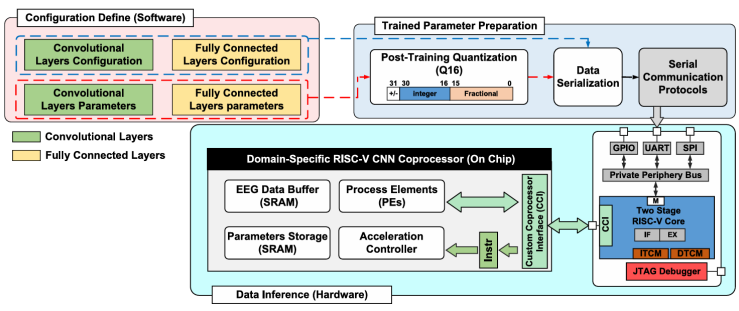


**Figure 2.** ASIC block diagram from [129].

In the context of portable EEG signal processing, FPGAs have become a mainstream choice for research. The reason is that the architectural design of FPGAs is very suitable for this purpose, and their unique flexibility and efficiency provide high computing performance with low power consumption. In [127], the authors compared three platforms—an Intel Core i7-7500U CPU, an NVIDIA Jetson Nano GPU, and a Xilinx Ultrascale+ ZU7EV FPGA—for motor imagination tasks. As shown in Figure 3, in terms of FOM and energy consumption rankings, the FPGA may be the best solution, followed by the Jetson Nano.

| | CPU | FPGA | Jetson Nano | |
|---|---|---|---|---|
| Power consumption (W) | 13.22 | 1.47 | 5 | 10 |
| Inference time (ms) | 121.57 | 19.97 | 15.85 | 12.21 |
| FOM (W × s) | 1.61 | $29.36 \times 10^{-3}$ | $79.25 \times 10^{-3}$ | $122.1 \times 10^{-3}$ |
| Memory footprint (Mb) | 476 | 6.37 | 415 | |
| Test accuracy | 81% | 77% | 81% | |

**Figure 3.** Results on different platforms from [127].

In [9], the authors proposed a software and hardware co-design solution for a lightweight CNN. As shown in Figure 4, they proposed another FPGA acceleration implementation. Unlike [128], the calculation results of each layer are transmitted back to the ARM processor through the AXI bus. This implementation method can flexibly configure various parameters of the CNN to adapt to rapidly evolving CNN technology but at the cost of increased data pressure. The results of the eye-state recognition task were evaluated on three platforms. As shown in Figure 5, FPGA has obvious advantages in weight, cost, and power consumption. In contrast, it is slightly less efficient than a high-performance PC in terms of energy consumption.

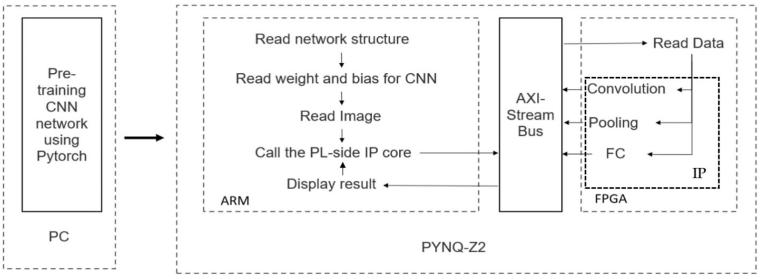


**Figure 4.** FPGA block diagram from [9].

| Hardware Platform | CPU | Accelerator | Memory | Inference Time | Accuracy | Power | Power Efficiency | Price | Weight |
|---|---|---|---|---|---|---|---|---|---|
| Laptop | Intel Core i5-7300HQ @ 2.50 GHz | GTX 1050Ti @ 1620 Mhz | 8 GB DDR4 @2666 MHz | 0.003 s | 89.92% | 131.1 W | 2.5 | ~$750 | 3.61 kg |
| PYNQ-Z2 | ARM Dual-core Cortex A9 @ 650 MHz | - | 512 MB DDR3 @525 MHz | 5.1 s | 89.85% | 1.9 W | 0.1 | ~$177 | 0.22 kg |
| PYNQ-Z2 | ARM Dual-core Cortex A9 @ 650 MHz | FPGA @ 100 Mhz | 512 MB DDR3 @525 MHz | 0.22 s | 89.69% | 3.1 W | 1.46 | ~$177 | 0.22 kg |

**Figure 5.** Results on different platforms from [9].

The authors of [22] introduced BioCNN, which was designed for emotion classification tasks. BioCNN was implemented on a Digilent Atlys board with a low-cost Spartan-6 FPGA and tested using the FEXD and DEAP datasets. It achieved an energy efficiency of 11 GOPS/W and a latency of less than 1 ms, significantly below the standard 150 ms threshold for human–computer interaction and approaching the performance of software-based solutions. Compared with the GPU, the FPGA not only accelerates the CNN inference process by processing a large number of multiply-accumulate (MAC) operations in parallel but also optimizes memory access and data flow processing through customized hardware

logic design. Because the FPGA's processing elements (PEs) have local storage and control logic, they can communicate directly with each other with minimal memory bandwidth requirements, significantly increasing data reuse. This feature is particularly critical for the low-power and efficient computing requirements of portable EEG devices.

In contrast, although GPUs perform well in the field of general computing, in the literature we surveyed, most of them were implemented in high-power PCs. Additionally, their low customization capabilities limit their application on portable devices. ASICs, on the other hand, although capable of delivering extreme performance and efficiency for specific applications, are generally only suitable for specific applications in mass production due to their high development costs and lack of flexibility. Therefore, FPGAs have become the preferred hardware platform for CNN acceleration in portable EEG signal-processing devices due to their low power consumption, flexibility, and sufficient computing power.

Table 3 summarizes the state-of-the-art CNN hardware solutions discussed in this section.

**Table 3.** List of reviewed CNN hardware techniques.

| Method | Authors | Work Description |
| --- | --- | --- |
| Weight Pruning | [51,52] | Hessian matrix of the loss function |
| | [53,54] | Three-step method |
| | [57] | Frequency-domain approach |
| | [51] | Variable compression rates |
| | [58] | GMM |
| | [59] | Energy reduction with accuracy loss |
| Structural Pruning | [60–62] | Criterion-based pruning |
| | [63] | Ranking-based pruning |
| | [64,65] | Optimization-based pruning |
| | [66–68] | Whole-layer pruning methods |
| | [69–71] | Automatically identifies and prunes |
| Quantization | [72] | Characterized by even step sizes |
| | [73] | Represented by logarithmic distributions |
| | [75,76] | Post-training quantization |
| | [77–79] | Quantization-aware training |
| | [80–82] | Binary quantization networks |
| | [72,82–85] | Optimized binarization techniques |
| LRM Decomposition | [88] | Decomposes the product of weight matrices and input data |
| | [89] | Sparsity: Maintaining a lower rank for less critical neurons |
| | [90] | Channel-wise SVD decomposition by dividing kernels into two layers |
| | [91] | Low-rank approximation without invoking SVD |
| | [92] | Joint matrix factorization scheme |
| Tensorized Decomposition | [93] | Tucker decomposition |
| | [94,95] | CANDECOMP/PARAFAC (CP) |
| | [96–98] | Tensor train decomposition |
| | [99] | Tensor ring decomposition |
| Knowledge Distillation | [101,103–105] | Offline distillation |
| | [106–108] | Online distillation |
| | [109,110] | Self-knowledge distillation |
| NA Search Space | [113] | NASNet |
| | [114] | ENAS |
| | [115] | AutoDispNet |
| | [116] | MnasNet |
| NA Search Algorithm | [112] | NAS-RL |
| | [117] | MetaQnn-RL |
| | [119] | Optimizing neural structures |
| NA Search Performance | [120] | Progressive NAS |
| | [121] | ReNAS |
| | [122] | One-shot NAS |
| | [123,124] | Hardware-aware NAS |

"NA" means neural architecture. "LRM" means low-rank matrix.

## 4. Challenges and Opportunities

### *4.1. Standardized Evaluation Paradigms*

Due to the different EEG application tasks, different indicators are needed to evaluate their performance. These indicators are diverse and complex. Due to the lack of standardized performance evaluation paradigms, not all EEG signal-processing hardware solutions provide the same indicators. Examples of these indicators include the following:

- **Wearability:** Wearability is an important feature in tasks that require constant, unobtrusive, real-time monitoring, such as sleep tracking or emotion classification. Uncomfortable or bulky devices may discourage users from consistently wearing them, potentially affecting data quality and continuity.
- **Energy Efficiency and Area**: Different hardware architectures cannot usually be directly compared. For example, FPGAs typically consume more power than ASICs. Moreover, the power consumption of an FPGA largely depends on its evaluation board. In other words, even the same FPGA design may lack comparability. For example, Ref. [131] introduced a normalized energy metric for comparing the power consumption of different technologies. The formula is as follows:

$$Normalized_{Energy} = \frac{Power \times ExecTime}{N_{Class}\left(\frac{V_{DD}}{V_{ref}}\right)^2} \quad (6)$$

  where the unit of normalized energy is $nJ$ and $N_{Class}$ is the number of classes.

- **Latency**: Latency is the time it takes for the system to produce classification results after receiving an EEG epoch. For instance, iMotor imagery tasks are highly dynamic, so shorter delays are desirable in such tasks. Subjects are less likely to maintain high levels of attention and exhibit consistent responses over longer time intervals. This contrasts sharply with seizure detection devices based on EEG, where longer detection time windows are more effective, as relevant seizures become sparser over time [132].

Standardized metrics and evaluation paradigms help to fairly compare different hardware solutions, promote healthy competition, and accelerate technological progress. For techniques such as compression, quantization, and hardware acceleration, standardized metrics help to explore the best efficiency for specific applications.

### *4.2. High-Quality EEG Datasets*

Acquiring high-quality EEG datasets is a formidable challenge, significantly impacting the robustness assessment of systems. For DL networks, such as CNNs, that require large and diverse data, the quantity and quality of EEG data affect their training results. Unfortunately, most available EEG datasets lack the approval needed as benchmark resources to validate EEG-based classifiers in medical settings. Therefore, standardization of data collection or involving more subjects in the dataset development process can promote the development of EEG signal-processing technology, both in software and hardware.

Therefore, datasets must provide raw EEG data along with comprehensive data acquisition details. For instance, [133,134] used bandpass Hamming windows as filters. In contrast, [135] employed built-in Sinc filters to eliminate frequency components above specified thresholds, and [131] used a high-pass filter to remove unwanted event codes and noise channels from the raw input data. Equally important is defining stimulus types before EEG data collection, with common choices being auditory, visual, and audiovisual mixed stimuli. However, in a particular task, the choice of stimuli is determined subjectively by the researcher, making it difficult to replicate the classification results. Stimulus selection may become one of the challenges of EEG signal acquisition.

Reporting other subject-related information, such as race, occupation, personality traits, and physiological conditions, as well as comprehensive experimental setup details, is crucial for identifying external influences. Reference sensors in EEG data collection are also vital for anchoring data points. Some datasets [136] use single-modal approaches (only

EEG data), but multimodal datasets [133,137], including ECG, facial video clips, peripheral physiological signals, and infrared spectra, offer a more comprehensive perspective and additional classification strategies.

In particular, CNNs can seamlessly integrate these multi-modal data sources to enhance the model's generalization ability. In conclusion, future EEG-based datasets for standardized benchmarks should encompass multi-modality, universality, consistent labeling, and preprocessing libraries, among other features.

### 4.3. Embedded Preprocessing

As previously mentioned, the removal of artifacts can directly impact the performance of CNNs. These artifacts often mask meaningful neural activity. For instance, power-line noise can obscure cortical activity around 50 Hz (or 60 Hz), while electrode drift can disrupt slow cortical potentials. When eye- or muscle-related activity occurs simultaneously with external stimulation, these artifacts may be mistaken for cortical activity [138]. This is a common situation in EEG data collection experiments. However, artifact removal can sometimes introduce unintended side effects, such as the multiplicative offset triggered by high-pass filters, which may occasionally be misinterpreted as neural activity [139]. Filter-based preprocessing alone cannot eliminate all artifacts from EEG data. The design of EEG-based hardware systems also needs to be combined with other efficient preprocessing techniques. While the ICA-based techniques mentioned earlier are effective, they are not entirely hardware-friendly and may not comprehensively address all types of artifacts. For instance, ICA cannot distinguish artifacts originating from electrode-specific or sensor-specific sources [140]. Therefore, finding and developing solutions that can effectively handle various artifacts while maximizing the use of hardware characteristics remains a key direction for achieving real-time classification of EEG signals.

### 4.4. High-Resolution Feature Extraction

The concept of EEG frequency bands, although valuable in engineering applications for enhancing classification accuracy, lacks comprehensive neurophysiological underpinning. Relying on PSD as a proxy for extracting EEG signals representing brain activity is an effective but somewhat rudimentary approach. Integrating CFC features into EEG-based hardware classifiers presents a promising alternative for wearable devices. This approach has the potential to enhance robustness and reduce memory requirements by emphasizing connectivity graphs rather than storing complete raw EEG datasets [141].

In the field of frequency-domain features, another avenue for improvement involves systematically incorporating spatial asymmetries. As discussed in Section 2.2.4, extending this practice to other EEG electrode configurations can further refine our understanding of EEG wave patterns and their impact on CNN-based EEG classifiers in both software and hardware implementations.

## 5. Conclusions

This review focuses on preprocessing, feature extraction hardware solutions, and acceleration techniques for CNNs analyzing EEG signals. It encompasses a wide range of applications, including emotion classification, motor imagery, seizure detection, and sleep monitoring. This review begins with a detailed analysis of hardware implementation techniques for EEG signal preprocessing and five types of feature extraction: frequency-based, time-based, time-frequency, spatial features, and raw signals. Subsequently, efficient CNN solutions based on EEG signals are elaborated, including pruning, quantization, tensor decomposition, knowledge distillation, and NAS. In the following sections, this review addresses the challenges and opportunities associated with EEG hardware accelerators, such as multi-modal universal datasets, standardized metrics, embedded preprocessing, and high-resolution feature extraction. Therefore, this article summarizes existing research results in the field of EEG signal-processing hardware implementation by reviewing the

current literature. It also identifies unresolved problems and research gaps and provides future research directions for other researchers.

**Author Contributions:** Conceptualization, Y.X. and S.O.; methodology, Y.X.; validation, Y.X.; formal analysis, Y.X.; investigation, Y.X.; writing—original draft preparation, Y.X.; writing—review and editing, Y.X. and S.O.; supervision, S.O. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | | | |
|---|---|---|---|
| ARRC | Auto-Regressive Reflection Coefficient | HOS | Higher-Order Spectral |
| APoZ | Average Percentage of Zeros | ICA | Independent Component Analysis |
| ASKI | Approximate Skewness Indicator | KDCL | Collaborative Learning Knowledge Distillation |
| BCI | Brain–Computer Interface | LFCC | Linear-Frequency Cepstral Coefficient |
| BNN | Binary Neural Networks | LDA | Linear Discriminant Analysis |
| CP | CANDECOMP/PARAFAC | LBP | Local Binary Pattern |
| CFC | Cross-Frequency Coupling | LPP | Late Positive Potentials |
| CNN | Convolutional Neural Network | LSL | Layer Selective Learning |
| CSP | Common Spatial Pattern | LSL | Layer Selective Learning |
| CWT | Continuous Wavelet Transform | MFCC | Mel-Frequency Cepstral Coefficient |
| CORDIC | Coordinate Rotation Digital Computer | MD | Mahalanobis Distance |
| DCT | Discrete Cosine Transform | NAS | Neural Architecture Search |
| DE | Differential Entropy | IMF | Independent Mode Function |
| DTF | Directed Transfer Function | RMS | Root Mean Square |
| DWT | Discrete Wavelet Transform | Var | Variance |
| DNN | Deep Neural Network | SD | Standard Deviation |
| GPUs | Graphics Processing Units | NTSPP | Neural Time-Series Prediction Processing |
| GP | Grassberger–Procaccia | OBD | Optimal Brain Damage |
| GMM | Gaussian Mixture Model | OBS | Optimal Brain Surgeon |
| EEG | Electroencephalography | PSD | Power Spectral Density |
| EVD | Eigenvalue Decomposition | RNN | Recurrent Neural Network |
| EMD | Empirical Mode Decomposition | RL | Reinforcement Learning |
| FPGA | Field-Programmable Gate Array | RS | Random Search |
| FRL | Final Response Layer | SK | Skewness |
| FDSOI | Fully Depleted Silicon on Insulator | SE | Sample Entropy |
| HFD | Higuchi's Fractal Dimension | SCSSP | Scalp-Coupled Common Spatial Spectral Pattern |
| HHT | Hilbert–Huang Transform | SVD | Singular-Value Decomposition |
| HAF | Hybrid Adaptive Filtering | SFP | Soft Filter Pruning |
| SVM | Support Vector Machine | TT | Tensor Train |
| STFT | Short-Time Fourier Transform | WT | Wavelet Transform |
| TR | Tensor Ring | | |

## References

1. Craik, A.; He, Y.; Contreras-Vidal, J.L. Deep learning for electroencephalogram (EEG) classification tasks: A review. *J. Neural Eng.* **2019**, *16*, 031001. [CrossRef] [PubMed]
2. Lotte, F.; Bougrain, L.; Cichocki, A.; Clerc, M.; Congedo, M.; Rakotomamonjy, A.; Yger, F. A review of classification algorithms for EEG-based brain–computer interfaces: A 10 year update. *J. Neural Eng.* **2018**, *15*, 031005. [CrossRef]
3. Wei, Y.; Zhou, J.; Wang, Y.; Liu, Y.; Liu, Q.; Luo, J.; Wang, C.; Ren, F.; Huang, L. A review of algorithm & hardware design for AI-based biomedical applications. *IEEE Trans. Biomed. Circuits Syst.* **2020**, *14*, 145–163.
4. Nunez, P.L.; Srinivasan, R. *Electric Fields of the Brain: The Neurophysics of EEG*; Oxford University Press: Oxford, UK, 2006.
5. Aggarwal, S.; Chugh, N. Review of machine learning techniques for EEG based brain computer interface. *Arch. Comput. Methods Eng.* **2022**, *29*, 3001–3020. [CrossRef]
6. Lu, H.; Wang, M.; Sangaiah, A.K. Human emotion recognition using an EEG cloud computing platform. *Mob. Netw. Appl.* **2020**, *25*, 1023–1032. [CrossRef]
7. Dong, L.; Li, J.; Zou, Q.; Zhang, Y.; Zhao, L.; Wen, X.; Gong, J.; Li, F.; Liu, T.; Evans, A.C.; et al. WeBrain: A web-based brainformatics platform of computational ecosystem for EEG big data analysis. *NeuroImage* **2021**, *245*, 118713. [CrossRef] [PubMed]

8. Ghimire, D.; Kil, D.; Kim, S.H. A survey on efficient convolutional neural networks and hardware acceleration. *Electronics* **2022**, *11*, 945. [CrossRef]
9. Xie, Y.; Majoros, T.; Oniga, S. FPGA-Based Hardware Accelerator on Portable Equipment for EEG Signal Patterns Recognition. *Electronics* **2022**, *11*, 2410. [CrossRef]
10. Johannisson, T. Correlations between personality traits and specific groups of alpha waves in the human EEG. *PeerJ* **2016**, *4*, e2245. [CrossRef]
11. Radüntz, T.; Scouten, J.; Hochmuth, O.; Meffert, B. EEG artifact elimination by extraction of ICA-component features using image processing algorithms. *J. Neurosci. Methods* **2015**, *243*, 84–93. [CrossRef]
12. Narayan, R. *Encyclopedia of Biomedical Engineering*; Elsevier: Amsterdam, The Netherlands, 2018.
13. Stone, D.B.; Tamburro, G.; Fiedler, P.; Haueisen, J.; Comani, S. Automatic removal of physiological artifacts in EEG: The optimized fingerprint method for sports science applications. *Front. Hum. Neurosci.* **2018**, *12*, 96. [CrossRef] [PubMed]
14. Bhuvaneswari, P.; Kumar, J.S. Methods used for identifying EEG signal artifacts. In Proceedings of the International Conference on Computational Intelligence and Information Technology, Coimbatore, Tamilnadu, India, 2–3 March 2012; pp. 375–379.
15. Khatter, A.; Bansal, D.; Mahajan, R. Study of various automatic eeg artifact removal techniques. *Int. J. Res. Appl. Sci. Eng. Technol.* **2017**, *5*, 1027–1037. [CrossRef]
16. Saltuklaroglu, T.; Bowers, A.; Harkrider, A.W.; Casenhiser, D.; Reilly, K.J.; Jenson, D.E.; Thornton, D. EEG mu rhythms: Rich sources of sensorimotor information in speech processing. *Brain Lang.* **2018**, *187*, 41–61. [CrossRef]
17. Yang, C.J.; Fahier, N.; He, C.Y.; Li, W.C.; Fang, W.C. An ai-edge platform with multimodal wearable physiological signals monitoring sensors for affective computing applications. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Online, 10–21 October 2020; pp. 1–5.
18. Aslam, A.R.; Altaf, M.A.B. An 8 channel patient specific neuromorphic processor for the early screening of autistic children through emotion detection. In Proceedings of the 2019 IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 26–29 May; pp. 1–5.
19. Aslam, A.R.; Iqbal, T.; Aftab, M.; Saadeh, W.; Altaf, M.A.B. A10. 13 uJ/classification 2-channel deep neural network-based SoC for emotion detection of autistic children. In Proceedings of the 2020 IEEE Custom Integrated Circuits Conference (CICC), Boston, MA, USA, 22–25 March 2020; pp. 1–4.
20. Yarahuaman, J.C.R.; Huamaní-Navarrete, P.F. Design and Simulation of a Digital Filter in Hardware for EEG Signals Based on FPGA. In Proceedings of the 2020 IEEE ANDESCON, Quito, Ecuador, 13–16 October 2020; pp. 1–6.
21. Aslam, A.R.; Altaf, M.A.B. An on-chip processor for chronic neurological disorders assistance using negative affectivity classification. *IEEE Trans. Biomed. Circuits Syst.* **2020**, *14*, 838–851. [CrossRef]
22. Gonzalez, H.A.; Muzaffar, S.; Yoo, J.; Elfadel, I.A.M. An inference hardware accelerator for EEG-based emotion detection. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Online, 10–21 October 2020; pp. 1–5.
23. Yang, C.H.; Shih, Y.H.; Chiueh, H. An 81.6 μW FastICA Processor for Epileptic Seizure Detection. *IEEE Trans. Biomed. Circuits Syst.* **2014**, *9*, 60–71. [CrossRef]
24. Wu, D.; Han, X.; Yang, Z.; Wang, R. Exploiting transfer learning for emotion recognition under cloud-edge-client collaborations. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 479–490. [CrossRef]
25. Tao, X.; Chen, Z.; Xu, M.; Lu, J. Rebuffering optimization for DASH via pricing and EEG-based QoE modeling. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1549–1565. [CrossRef]
26. Chen, Y.H.; Chen, S.W.; Wei, M.X. A VLSI implementation of independent component analysis for biomedical signal separation using CORDIC engine. *IEEE Trans. Biomed. Circuits Syst.* **2020**, *14*, 373–381. [CrossRef] [PubMed]
27. Chen, Y.H.; Wang, S.P. Low-cost implementation of independent component analysis for biomedical signal separation using very-large-scale integration. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 3437–3441. [CrossRef]
28. Altaheri, H.; Muhammad, G.; Alsulaiman, M.; Amin, S.U.; Altuwaijri, G.A.; Abdul, W.; Bencherif, M.A.; Faisal, M. Deep learning techniques for classification of electroencephalogram (EEG) motor imagery (MI) signals: A review. *Neural Comput. Appl.* **2023**, *35*, 14681–14722. [CrossRef]
29. Knyazev, G.G.; Savostyanov, A.N.; Bocharov, A.V.; Tamozhnikov, S.S.; Kozlova, E.A.; Leto, I.V.; Slobodskaya, H.R. Cross-frequency coupling in developmental perspective. *Front. Hum. Neurosci.* **2019**, *13*, 158. [CrossRef]
30. Yang, J.; Sawan, M. From seizure detection to smart and fully embedded seizure prediction engine: A review. *IEEE Trans. Biomed. Circuits Syst.* **2020**, *14*, 1008–1023. [CrossRef]
31. Davis, P.; Creusere, C.D.; Tang, W. ASIC implementation of the cross frequency coupling algorithm for EEG signal processing. In Proceedings of the 2014 International Symposium on Integrated Circuits (ISIC), Marina Bay Sands, Singapore, 10–12 December 2014; pp. 248–251.
32. Rizal, A.; Hadiyoso, S.; Ramdani, A.Z. FPGA-Based Implementation for Real-Time Epileptic EEG Classification Using Hjorth Descriptor and KNN. *Electronics* **2022**, *11*, 3026. [CrossRef]
33. Richman, J.S.; Moorman, J.R. Physiological time-series analysis using approximate entropy and sample entropy. *Am. J. Physiol. Heart Circ. Physiol.* **2000**, *278*, H2039–H2049. [CrossRef]
34. Chen, C.; Da Silva, B.; Li, J.; Liu, C. Acceleration of Fast Sample Entropy Towards Biomedical Applications on FPGAs. In Proceedings of the 2022 International Conference on Field-Programmable Technology (ICFPT), Hong Kong, China, 5–9 December 2022; pp. 1–4.

35. Wickramasuriya, D.S.; Wijesinghe, L.P.; Mallawaarachchi, S. Seizure prediction using Hilbert Huang Transform on field programmable gate array. In Proceedings of the 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Orlando, FL, USA, 14–16 December 2015; pp. 933–937.
36. Zhang, L. Real-time feature extraction for multi-channel EEG signals time-frequency analysis. In Proceedings of the 2017 8th International IEEE/EMBS Conference on Neural Engineering (NER), Shanghai, China, 25–28 May 2017; pp. 493–496.
37. Beeraka, S.M.; Kumar, A.; Sameer, M.; Ghosh, S.; Gupta, B. Accuracy enhancement of epileptic seizure detection: A deep learning approach with hardware realization of STFT. *Circuits Syst. Signal Process.* **2022**, *41*, 461–484. [CrossRef]
38. Fang, W.C.; Wang, K.Y.; Fahier, N.; Ho, Y.L.; Huang, Y.D. Development and validation of an EEG-based real-time emotion recognition system using edge AI computing platform with convolutional neural network system-on-chip design. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2019**, *9*, 645–657. [CrossRef]
39. Ellawala, N.; Thayaparan, S. Hardware implementation of eeg classifier using lda. In Proceedings of the 2019 2nd International Conference on Bioinformatics, Biotechnology and Biomedical Engineering (BioMIC)-Bioinformatics and Biomedical Engineering, Yogyakarta, IN, USA, 12–13 September 2019; Volume 1, pp. 1–5.
40. Qassim, Y.T.; Cutmore, T.; James, D.; Rowlands, D. FPGA implementation of Morlet continuous wavelet transform for EEG analysis. In Proceedings of the 2012 International Conference on Computer and Communication Engineering (ICCCE), Kuala Lumpur, Malaysia, 3–5 July 2012; pp. 59–64.
41. Belwafi, K.; Ghaffari, F.; Djemal, R.; Romain, O. A hardware/software prototype of EEG-based BCI system for home device control. *J. Signal Process. Syst.* **2017**, *89*, 263–279. [CrossRef]
42. Kais, B.; Ghaffari, F.; Romain, O.; Djemal, R. An embedded implementation of home devices control system based on brain computer interface. In Proceedings of the 2014 26th International Conference on Microelectronics (ICM), Doha, Qatar, 14–17 December 2014; pp. 140–143.
43. Malekmohammadi, A.; Mohammadzade, H.; Chamanzar, A.; Shabany, M.; Ghojogh, B. An efficient hardware implementation for a motor imagery brain computer interface system. *Sci. Iran.* **2019**, *26*, 72–94. [CrossRef]
44. Duan, R.N.; Zhu, J.Y.; Lu, B.L. Differential entropy feature for EEG-based emotion classification. In Proceedings of the 2013 6th International IEEE/EMBS Conference on Neural Engineering (NER), San Diego, CA, USA, 6–8 November 2013; pp. 81–84.
45. Kaminski, M.J.; Blinowska, K.J. A new method of the description of the information flow in the brain structures. *Biol. Cybern.* **1991**, *65*, 203–210. [CrossRef]
46. Wang, Y.; Huang, Z.; McCane, B.; Neo, P. EmotioNet: A 3-D convolutional neural network for EEG-based emotion recognition. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–7.
47. Salama, E.S.; El-Khoribi, R.A.; Shoman, M.E.; Shalaby, M.A.W. EEG-based emotion recognition using 3D convolutional neural networks. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, *8*. [CrossRef]
48. Singh, K.; de Gyvez, J.P. Twenty years of near/sub-threshold design trends and enablement. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *68*, 5–11. [CrossRef]
49. Höppner, S.; Eisenreich, H.; Walter, D.; Scharfe, A.; Oefelein, A.; Schraut, F.; Schreiter, J.; Riedel, T.; Bauer, H.; Niebsch, R. Adaptive body bias aware implementation for ultra-low-voltage designs in 22FDX technology. *IEEE Trans. Circuits Syst. II Express Briefs* **2019**, *67*, 2159–2163. [CrossRef]
50. Gao, Z.; Li, Y.; Yang, Y.; Dong, N.; Yang, X.; Grebogi, C. A coincidence-filtering-based approach for CNNs in EEG-based recognition. *IEEE Trans. Ind. Inform.* **2019**, *16*, 7159–7167. [CrossRef]
51. Ding, X.; Zhou, X.; Guo, Y.; Han, J.; Liu, J. Global sparse momentum sgd for pruning very deep neural networks. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 573.
52. Hassibi, B.; Stork, D. Second order derivatives for network pruning: Optimal brain surgeon. *Adv. Neural Inf. Process. Syst.* **1992**, *5*.
53. Han, S.; Pool, J.; Tran, J.; Dally, W. Learning both weights and connections for efficient neural network. *Adv. Neural Inf. Process. Syst.* **2015**, *28*.
54. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* **2015**, arXiv:1510.00149.
55. Zhu, M.; Gupta, S. To prune, or not to prune: Exploring the efficacy of pruning for model compression. *arXiv* **2017**, arXiv:1710.01878.
56. Alford, S.; Robinett, R.; Milechin, L.; Kepner, J. Training behavior of sparse neural network topologies. In Proceedings of the 2019 IEEE High Performance Extreme Computing Conference (HPEC), Westin Hotel, Waltham, MA, USA, 24–26 September 2019; pp. 1–6.
57. Liu, Z.; Xu, J.; Peng, X.; Xiong, R. Frequency-domain dynamic pruning for convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2018**, *11*, 1051–1061.
58. Lee, E.; Hwang, Y. Layer-Wise Network Compression Using Gaussian Mixture Model. *Electronics* **2021**, *10*, 72. [CrossRef]
59. Yang, T.J.; Chen, Y.H.; Sze, V. Designing energy-efficient convolutional neural networks using energy-aware pruning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5687–5695.
60. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning filters for efficient convnets. *arXiv* **2016**, arXiv:1608.08710.
61. He, Y.; Kang, G.; Dong, X.; Fu, Y.; Yang, Y. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv* **2018**, arXiv:1808.06866.

62. Hu, H.; Peng, R.; Tai, Y.W.; Tang, C.K. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv* **2016**, arXiv:1607.03250.
63. Lin, M.; Ji, R.; Wang, Y.; Zhang, Y.; Zhang, B.; Tian, Y.; Shao, L. Hrank: Filter pruning using high-rank feature map. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1529–1538.
64. Luo, J.H.; Wu, J.; Lin, W. Thinet: A filter level pruning method for deep neural network compression. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5058–5066.
65. Yu, R.; Li, A.; Chen, C.F.; Lai, J.H.; Morariu, V.I.; Han, X.; Gao, M.; Lin, C.Y.; Davis, L.S. Nisp: Pruning networks using neuron importance score propagation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9194–9203.
66. Chen, S.; Zhao, Q. Shallowing deep networks: Layer-wise pruning based on feature representations. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 3048–3056. [CrossRef] [PubMed]
67. Elkerdawy, S.; Elhoushi, M.; Singh, A.; Zhang, H.; Ray, N. To filter prune, or to layer prune, that is the question. In Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December 2020.
68. Xu, P.; Cao, J.; Shang, F.; Sun, W.; Li, P. Layer pruning via fusible residual convolutional block for deep neural networks. *arXiv* **2020**, arXiv:2011.14356.
69. Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; Darrell, T. Rethinking the value of network pruning. *arXiv* **2018**, arXiv:1810.05270.
70. Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; Zhang, C. Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2736–2744.
71. Wang, Y.; Zhang, X.; Xie, L.; Zhou, J.; Su, H.; Zhang, B.; Hu, X. Pruning from scratch. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12273–12280.
72. Zhou, S.; Wu, Y.; Ni, Z.; Zhou, X.; Wen, H.; Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv* **2016**, arXiv:1606.06160.
73. Miyashita, D.; Lee, E.H.; Murmann, B. Convolutional neural networks using logarithmic data representation. *arXiv* **2016**, arXiv:1603.01025.
74. Fiesler, E.; Choudry, A.; Caulfield, H.J. Weight discretization paradigm for optical neural networks. In Proceedings of the Optical Interconnections and Networks, Hamburg, Germany, 14–15 March 1990; Volume 1281, pp. 164–173.
75. Wu, H.; Judd, P.; Zhang, X.; Isaev, M.; Micikevicius, P. Integer quantization for deep learning inference: Principles and empirical evaluation. *arXiv* **2020**, arXiv:2004.09602.
76. Banner, R.; Nahshan, Y.; Soudry, D. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
77. Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2704–2713.
78. Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Quantized neural networks: Training neural networks with low precision weights and activations. *J. Mach. Learn. Res.* **2017**, *18*, 6869–6898.
79. Zhou, Q.; Guo, S.; Qu, Z.; Guo, J.; Xu, Z.; Zhang, J.; Guo, T.; Luo, B.; Zhou, J. Octo:INT8 training with loss-aware compensation and backward quantization for tiny on-device learning. In Proceedings of the 2021 USENIX Annual Technical Conference (USENIX ATC 21), Online, 14–16 July 2021; pp. 177–191.
80. Courbariaux, M.; Bengio, Y.; David, J.P. Binaryconnect: Training deep neural networks with binary weights during propagations. *Adv. Neural Inf. Process. Syst.* **2015**, *28*.
81. Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or −1. *arXiv* **2016**, arXiv:1602.02830.
82. Rastegari, M.; Ordonez, V.; Redmon, J.; Farhadi, A. Xnor-net: Imagenet classification using binary convolutional neural networks. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 525–542.
83. Hou, L.; Yao, Q.; Kwok, J.T. Loss-aware binarization of deep networks. *arXiv* **2016**, arXiv:1611.01600.
84. Zhou, A.; Yao, A.; Guo, Y.; Xu, L.; Chen, Y. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv* **2017**, arXiv:1702.03044.
85. Qin, H.; Gong, R.; Liu, X.; Shen, M.; Wei, Z.; Yu, F.; Song, J. Forward and backward information retention for accurate binary neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2250–2259.
86. Denil, M.; Shakibi, B.; Dinh, L.; Ranzato, M.; De Freitas, N. Predicting parameters in deep learning. *Adv. Neural Inf. Process. Syst.* **2013**, *26*.
87. Klema, V.; Laub, A. The singular value decomposition: Its computation and some applications. *IEEE Trans. Autom. Control.* **1980**, *25*, 164–176. [CrossRef]
88. Xue, J.; Li, J.; Yu, D.; Seltzer, M.; Gong, Y. Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 6359–6363.
89. Swaminathan, S.; Garg, D.; Kannan, R.; Andres, F. Sparse low rank factorization for deep neural network compression. *Neurocomputing* **2020**, *398*, 185–196. [CrossRef]

90. Zhang, X.; Zou, J.; He, K.; Sun, J. Accelerating very deep convolutional networks for classification and detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 1943–1955. [CrossRef]
91. Yang, H.; Tang, M.; Wen, W.; Yan, F.; Hu, D.; Li, A.; Li, H.; Chen, Y. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 678–679.
92. Chen, S.; Zhou, J.; Sun, W.; Huang, L. Joint matrix decomposition for deep convolutional neural networks compression. *Neurocomputing* **2023**, *516*, 11–26. [CrossRef]
93. Kim, Y.D.; Park, E.; Yoo, S.; Choi, T.; Yang, L.; Shin, D. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv* **2015**, arXiv:1511.06530.
94. Lebedev, V.; Ganin, Y.; Rakhuba, M.; Oseledets, I.; Lempitsky, V. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *arXiv* **2014**, arXiv:1412.6553.
95. Astrid, M.; Lee, S.I. Cp-decomposition with tensor power method for convolutional neural networks compression. In Proceedings of the 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), Jeju Island, Republic of Korea, 13–16 February 2017; pp. 115–118.
96. Yang, Y.; Krompass, D.; Tresp, V. Tensor-train recurrent neural networks for video classification. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 3891–3900.
97. Yin, M.; Sui, Y.; Liao, S.; Yuan, B. Towards efficient tensor decomposition-based dnn model compression with optimization framework. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 10674–10683.
98. Wang, D.; Zhao, G.; Li, G.; Deng, L.; Wu, Y. Compressing 3DCNNs based on tensor train decomposition. *Neural Netw.* **2020**, *131*, 215–230. [CrossRef]
99. Li, N.; Pan, Y.; Chen, Y.; Ding, Z.; Zhao, D.; Xu, Z. Heuristic rank selection with progressively searching tensor ring network. In *Complex & Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 1–15.
100. Phan, A.H.; Sobolev, K.; Sozykin, K.; Ermilov, D.; Gusak, J.; Tichavský, P.; Glukhov, V.; Oseledets, I.; Cichocki, A. Stable low-rank tensor decomposition for compression of convolutional neural network. In Proceedings of the Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XXIX 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 522–539.
101. Polino, A.; Pascanu, R.; Alistarh, D. Model compression via distillation and quantization. *arXiv* **2018**, arXiv:1802.05668.
102. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
103. Fukuda, T.; Suzuki, M.; Kurata, G.; Thomas, S.; Cui, J.; Ramabhadran, B. Efficient Knowledge Distillation from an Ensemble of Teachers. In Proceedings of the Interspeech, Stockholm, Sweden, 20–24 August 2017; pp. 3697–3701.
104. Nayak, G.K.; Mopuri, K.R.; Shaj, V.; Radhakrishnan, V.B.; Chakraborty, A. Zero-shot knowledge distillation in deep networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 4743–4751.
105. Li, H.T.; Lin, S.C.; Chen, C.Y.; Chiang, C.K. Layer-level knowledge distillation for deep neural network learning. *Appl. Sci.* **2019**, *9*, 1966. [CrossRef]
106. Jin, X.; Peng, B.; Wu, Y.; Liu, Y.; Liu, J.; Liang, D.; Yan, J.; Hu, X. Knowledge distillation via route constrained optimization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1345–1354.
107. Guo, Q.; Wang, X.; Wu, Y.; Yu, Z.; Liang, D.; Hu, X.; Luo, P. Online knowledge distillation via collaborative learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11020–11029.
108. Walawalkar, D.; Shen, Z.; Savvides, M. Online ensemble model compression using knowledge distillation. In Proceedings of the Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XIX 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 18–35.
109. Ji, M.; Shin, S.; Hwang, S.; Park, G.; Moon, I.C. Refine myself by teaching myself: Feature refinement via self-knowledge distillation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 10664–10673.
110. Vu, D.Q.; Le, N.; Wang, J.C. Teaching yourself: A self-knowledge distillation approach to action recognition. *IEEE Access* **2021**, *9*, 105711–105723. [CrossRef]
111. Stewart, R.; Nowlan, A.; Bacchus, P.; Ducasse, Q.; Komendantskaya, E. Optimising hardware accelerated neural networks with quantisation and a knowledge distillation evolutionary algorithm. *Electronics* **2021**, *10*, 396. [CrossRef]
112. Zoph, B.; Le, Q.V. Neural architecture search with reinforcement learning. *arXiv* **2016**, arXiv:1611.01578.
113. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8697–8710.
114. Pham, H.; Guan, M.; Zoph, B.; Le, Q.; Dean, J. Efficient neural architecture search via parameters sharing. In Proceedings of the International Conference on Machine Learning, Stockholmsmässan, Stockholm Sweden, 10–15 July 2018; pp. 4095–4104.
115. Saikia, T.; Marrakchi, Y.; Zela, A.; Hutter, F.; Brox, T. Autodispnet: Improving disparity estimation with automl. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1812–1823.

116. Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; Le, Q.V. Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2820–2828.
117. Baker, B.; Gupta, O.; Naik, N.; Raskar, R. Designing neural network architectures using reinforcement learning. *arXiv* **2016**, arXiv:1611.02167.
118. Stanley, K.O.; Miikkulainen, R. Evolving neural networks through augmenting topologies. *Evol. Comput.* **2002**, *10*, 99–127. [CrossRef]
119. Real, E.; Aggarwal, A.; Huang, Y.; Le, Q.V. Aging evolution for image classifier architecture search. In Proceedings of the AAAI Conference on Artificial Intelligence, Hilton Hawaiian Village, Honolulu, HI, USA, 27 January–1 February 2019; Volume 2, p. 2.
120. Liu, C.; Zoph, B.; Neumann, M.; Shlens, J.; Hua, W.; Li, L.J.; Fei-Fei, L.; Yuille, A.; Huang, J.; Murphy, K. Progressive neural architecture search. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 19–34.
121. Xu, Y.; Wang, Y.; Han, K.; Tang, Y.; Jui, S.; Xu, C.; Xu, C. Renas: Relativistic evaluation of neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4411–4420.
122. Brock, A.; Lim, T.; Ritchie, J.M.; Weston, N. Smash: One-shot model architecture search through hypernetworks. *arXiv* **2017**, arXiv:1708.05344.
123. Cai, H.; Gan, C.; Wang, T.; Zhang, Z.; Han, S. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv* **2019**, arXiv:1908.09791.
124. Xia, X.; Xiao, X.; Wang, X.; Zheng, M. Progressive automatic design of search space for one-shot neural architecture search. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2022; pp. 2455–2464.
125. Zhang, L.L.; Yang, Y.; Jiang, Y.; Zhu, W.; Liu, Y. Fast hardware-aware neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 692–693.
126. Liu, P.; Wu, B.; Ma, H.; Seok, M. MemNAS: Memory-efficient neural architecture search with grow-trim learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2108–2116.
127. Pacini, F.; Pacini, T.; Lai, G.; Zocco, A.M.; Fanucci, L. Design and Evaluation of CPU-, GPU-, and FPGA-Based Deployment of a CNN for Motor Imagery Classification in Brain-Computer Interfaces. *Electronics* **2024**, *13*, 1646. [CrossRef]
128. Ma, X.; Zheng, W.; Peng, Z.; Yang, J. Fpga-based rapid electroencephalography signal classification system. In Proceedings of the 2019 IEEE 11th International Conference on Advanced Infocomm Technology (ICAIT), Jinan, China, 18–20 October 2019; pp. 223–227.
129. Lee, S.Y.; Hung, Y.W.; Chang, Y.T.; Lin, C.C.; Shieh, G.S. RISC-V CNN coprocessor for real-time epilepsy detection in wearable application. *IEEE Trans. Biomed. Circuits Syst.* **2021**, *15*, 679–691. [CrossRef]
130. Huang, Y.D.; Wang, K.Y.; Ho, Y.L.; He, C.Y.; Fang, W.C. An edge AI system-on-chip design with customized convolutional-neural-network architecture for real-time EEG-based affective computing system. In Proceedings of the 2019 IEEE Biomedical Circuits and Systems Conference (BioCAS), Nara, Japan, 17–19 October 2019; pp. 1–4.
131. Gonzalez, H.A.; George, R.; Muzaffar, S.; Acevedo, J.; Hoeppner, S.; Mayr, C.; Yoo, J.; Fitzek, F.H.; Elfadel, I.M. Hardware acceleration of EEG-based emotion classification systems: A comprehensive survey. *IEEE Trans. Biomed. Circuits Syst.* **2021**, *15*, 412–442. [CrossRef] [PubMed]
132. Yoo, J.; Yan, L.; El-Damak, D.; Altaf, M.B.; Shoeb, A.; Yoo, H.J.; Chandrakasan, A. An 8-channel scalable EEG acquisition SoC with fully integrated patient-specific seizure classification and recording processor. In Proceedings of the 2012 IEEE International Solid-State Circuits Conference, San Francisco, CA, USA, 19–23 February 2012; pp. 292–294.
133. Koelstra, S.; Muhl, C.; Soleymani, M.; Lee, J.S.; Yazdani, A.; Ebrahimi, T.; Pun, T.; Nijholt, A.; Patras, I. Deap: A database for emotion analysis; using physiological signals. *IEEE Trans. Affect. Comput.* **2011**, *3*, 18–31. [CrossRef]
134. Zhao, K.; Xu, D. Food image-induced discrete emotion recognition using a single-channel scalp-EEG recording. In Proceedings of the 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Suzhou China, 19–21 October 2019; pp. 1–6.
135. Alakus, T.B.; Gonen, M.; Turkoglu, I. Database for an emotion recognition system based on EEG signals and various computer games–GAMEEMO. *Biomed. Signal Process. Control* **2020**, *60*, 101951. [CrossRef]
136. Goldberger, A.L.; Amaral, L.A.; Glass, L.; Hausdorff, J.M.; Ivanov, P.C.; Mark, R.G.; Mietus, J.E.; Moody, G.B.; Peng, C.K.; Stanley, H.E. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation* **2000**, *101*, e215–e220. [CrossRef]
137. Park, C.Y.; Cha, N.; Kang, S.; Kim, A.; Khandoker, A.H.; Hadjileontiadis, L.; Oh, A.; Jeong, Y.; Lee, U. K-EmoCon, a multimodal sensor dataset for continuous emotion recognition in naturalistic conversations. *Sci. Data* **2020**, *7*, 293. [CrossRef]
138. Yeung, N.; Bogacz, R.; Holroyd, C.B.; Cohen, J.D. Detection of synchronized oscillations in the electroencephalogram: An evaluation of methods. *Psychophysiology* **2004**, *41*, 822–832. [CrossRef] [PubMed]
139. Tanner, D.; Morgan-Short, K.; Luck, S.J. How inappropriate high-pass filters can produce artifactual effects and incorrect conclusions in ERP studies of language and cognition. *Psychophysiology* **2015**, *52*, 997–1009. [CrossRef] [PubMed]

140. Viola, F.C.; Debener, S.; Thorne, J.; Schneider, T.R. Using ICA for the analysis of multi-channel EEG data. In *Simultaneous EEG and fMRI: Recording, Analysis, and Application: Recording, Analysis, and Application*; Oxford Academic: Oxford, UK, 2010; pp. 121–133.
141. Gwon, D.; Ahn, M. Alpha and high gamma phase amplitude coupling during motor imagery and weighted cross-frequency coupling to extract discriminative cross-frequency patterns. *NeuroImage* **2021**, *240*, 118403. [CrossRef] [PubMed]