National Research University Higher School of Economics
Faculty of computer science

Lebedev Mikhail

Towards few-shot learning on small-world neuromorphic hardware

Master's Thesis

Scientific supervisor:
Dr. Denis Moskvin

Scientific consultant:
Professor Melika Payvand

Reviewer:
Dr. Advait Madhavan

Saint-Petersburg
2023

Contents

Abstracts

This work represents a stride toward bridging the gap between Neural Architecture Search (NAS) and neuromorphic computing, underscoring the exciting possibilities for the further refinement and application of NAS methods for neuromorphic hardware. A tailored NAS approach for spiking neural networks is introduced in response to the requirement for simplified and efficient neural network architectures for this innovative technology. Our focus is on low power and miniaturized speech processing on the edge, utilizing the Spiking Heidelberg Digits (SHD) dataset. This dataset comprises approximately 10,000 high-quality studio recordings of spoken digits from 0 to 9 in both German and English, offering diverse data for model training. While the optimized architectures achieved an accuracy of 0.58, they fell short of state-of-the-art models due to a smaller set of trainable parameters and a smaller set of operations. However, when compared based on accuracy, this approach systematically identified superior architecture designs from the search space, situating them within the 90th percentile of a pool of 200 randomly generated architectures. Nevertheless, the models demonstrated efficiency and efficacy in handling tasks suitable for the neuromorphic paradigm, affirming the potential of this approach.

Keywords: neural architecture search, neuromorpic computing, automatic speech recognition, Differentiable Architecture Search

Эта работа представляет собой шаг к преодолению разрыва между алгоритмами поиска нейронной архитектуры (NAS) и нейроморфными вычислениями, подчеркивая захватывающие возможности для дальнейшего совершенствования и применения методов NAS для нейроморфного оборудования. В ответ на потребность в упрощенных и эффективных архитектурах нейронных сетей для этой инновационной технологии, представлен адаптированный подход к поиску нейронной архитектуры для нейронных сетей со спайками. Нацеливаясь на обработку речи на оборудовании с низким энергопотреблением,

использовался набор данных Spiking Heidelberg Digits (SHD). Этот набор данных состоит из примерно 10 000 высококачественных студийных записей произнесенных цифр от 0 до 9 как на немецком, так и на английском языках, предоставляя разнообразные данные для обучения модели. Хотя полученные архитектуры достигли точности 0,58, они отстают от state-of-the-art моделей из-за меньшего набора обучаемых параметров и небольшого набора возможных операций. Однако, при сравнении на основе точности, этот подход систематически определял оптимальные дизайны архитектур нейронных сетей из имеющегося пространства поиска, помещая их в 90-й процентиль пула из 200 случайно сгенерированных архитектур. Таким образом, модели продемонстрировали эффективность и действенность в решении задач, подходящих для нейроморфной парадигмы, подтверждая потенциал разработанного подхода.

Ключевые слова: поиск архитектур нейронных сетей, нейроморфные вычисления, автоматическое распознование речи, дифференцируемый поиск архитектур

Introduction

Neuromorphic computing represents a burgeoning frontier in the realm of computational systems design, drawing inspiration from the structure, function, and adaptability of biological brains. This innovative approach aspires to create artificial systems which mimic the characteristics and capabilities of biological neural systems. The introduction of neuromorphic hardware, due to its energy efficiency brought by locality and sparsity, offers exciting potential for enhancing machine learning algorithms. These advantages enable more economical solutions that are desirable for edge applications, transforming the landscape of computational solutions. The focus of this thesis is on keyword spotting, an immediate application that demonstrates the potential of these neuromorphic solutions in real-world scenarios [55].

Despite its promise, the emergent state of this technology brings with it specific challenges, most notably, the requirement for neural network architectures that are simultaneously simplified and efficient. In response to this demand, we present a novel architecture search algorithm, specifically adapted for spiking neural networks.

Our study sets the stage for a fresh approach to machine learning within the context of neuromorphic hardware, introducing simple yet powerful neural network architectures capable of managing complex auditory signal processing tasks. The flexibility and adaptability of this architecture search algorithm hold the potential to accelerate advancements across a wide array of edge applications, e.g., ranging from intelligent home devices to sophisticated healthcare systems.

In this study, we aim to design a novel algorithm for Neural Architecture Search (NAS), explicitly accounting for some of the inherent constraints of neuromorphic hardware, such as the range of viable operations. Our objectives are three-fold:

Firstly, we intend to select and tailor a suitable NAS algorithm to optimize neural architectures for neuromorphic hardware.

Subsequently, we will test the resultant neural network on spike-encoded speech signals to gauge its performance and efficacy.

Lastly, we will perform an analysis comparing the identified optimal neural network architecture with randomly selected ones within the same search space. This comparative study will elucidate the relative merits of our selected architecture, demonstrating the effectiveness of our tailored NAS algorithm in optimizing for neuromorphic hardware constraints.

This research aims to come closer to bridging the gap between NAS and neuromorphic computing, leveraging the synergistic benefits of both areas to go towards enhancing the design and performance of spiking neural networks.

1 Neuromorphic computing

1.1 Neuromorphic hardware

Neuromorphic computing, initially introduced by Carver Mead in the late 1980s [43], revolves around the objective of enhancing computational efficiency by incorporating principles of neuroscience in hardware design. The core component of neuromorphic systems is the artificial neuron, designed to mirror biological neurons in the brain. These artificial neurons are interconnected via synapses, tasked with signal transmission between neurons.

Neuromorphic hardware brings several distinct advantages over conventional computing systems. Foremost among these are the integration of memory and computation, which reduces data movement, hence decreasing energy consumption and latency. Furthermore, the event-driven characteristic of neuromorphic systems facilitates real-time processing and quick responsiveness to input alterations [29].

However, certain challenges accompany the transition to neuromorphic hardware. As a developing technology, creating efficient algorithms and software that can fully harness these novel architectures is an ongoing area of research. Additionally, training neuromorphic systems presents its own set of complexities due to their discrete, event-driven nature.

With conventional computing paradigms like Moore's Law and Dennard scaling reaching their natural limitations, neuromorphic computing is increasingly recognized as a promising avenue for further innovation. It has evolved from its origins in analogue-digital representations of brain-inspired computation to include a wide range of hardware configurations, fueled by significant initiatives such as the DARPA Synapse project [1] and the European Union's Human Brain Project [2].

Neuromorphic computers are distinguished from their von Neumann counterparts through architecture and functionality inspired by biological brains, comprised of a network of neurons and synapses [56]. In stark contrast to the separate processing and memory units of von Neumann systems, neuro-

[1] https://www.darpa.mil/program/systems-of-neuromorphic-adaptive-plastic-scalable-electronics
[2] https://www.humanbrainproject.eu/en/

morphic computers blend these functions within the network of neurons and synapses. Information is encoded via the timing, magnitude, and shape of input spikes, a stark contrast to the numerical representation of binary values within traditional systems. This dynamic conversion between spikes and binary values presents a fascinating area of ongoing research in neuromorphic computing.

Several operational distinctions make neuromorphic computers a transformative paradigm. First, neuromorphic computers operate in a highly parallel fashion, allowing simultaneous operation of all neurons and synapses, although the computations they perform are simpler than those in parallel von Neumann systems. Second, the merging of processing and memory in neuromorphic hardware mitigates the von Neumann bottleneck, optimizing throughput [29]. This also minimizes energy consumption, a major advantage in light of growing energy demands in computing. Neuromorphic computers also boast inherent scalability, enabling increasingly larger networks with the addition of neuromorphic chips. This feature has been successfully demonstrated in hardware systems like SpiNNaker [42] [77] and Loihi [41]. The event-driven computation of neuromorphic computers, leveraging temporal sparsity, enables highly efficient processing [47] [75]. Finally, a stochastic approach allows for inbuilt randomness akin to biological neuron firing.

The low-power operation, massively parallel nature, and intrinsic suitability for neural network-style computation position neuromorphic computing as a prime candidate for energy-constrained applications and edge computing. Nevertheless, many aspects of biological brains that may influence computation, such as the role of Glial cells or the level of abstraction in neuromorphic systems, remain open areas of research [61] [32] [54].

Several large-scale neuromorphic systems have been developed for various purposes like the Human Brain Project's SpiNNaker [42] and BrainScaleS [78], and the Tianjic chip [71]. Industry giants such as IBM and Intel have also developed neuromorphic systems like TrueNorth [76] and Loihi [41].

Further research in new types of materials for neuromorphic implementations, like phase-change, ferroelectric, non-filamentary, topological insulators, or channel-doped biomembranes, heralds a promising future for the field [17]

[9] [44], thanks to the multi-state, internal dynamics and dense non-volatile properties.

Despite the profound advancements in neuromorphic hardware, successful implementation of neuromorphic systems mandates the development of neuromorphic algorithms and applications. As research advances, it is vital to connect these computational characteristics with neuromorphic algorithms, driving hardware design and exploring novel applications in computer science and computational science. Neuromorphic computing might play a significant role in shaping the future of computation, promising unparalleled innovation as we move toward more efficient, brain-inspired computing paradigms.

Building upon the theories and research of computer science, neuroscience, and bioengineering, neuromorphic computing manifests as a multidisciplinary endeavor that seeks to distill the brain's computational principles and manifest them into artificial systems.

1.2 Neuromorphic algorithms and applications

The principal constituents of neuromorphic systems, spiking neural networks (SNNs), stand apart from their traditional artificial neural network counterparts due to their biological fidelity and emphasis on temporal dynamics. Instead of passing information through layers synchronously, as seen in conventional networks, SNNs propagate information asynchronously [60]. Each neuron and synapse possesses unique temporal delays, leading to the arrival of information at differing times.

SNNs encompass neurons that integrate charge over time, derived from either environmental input or internal communication. Once a neuron's charge reaches a threshold value, it fires, distributing communications through its outgoing synapses. These neurons may also exhibit leakage, where a charge that doesn't cross the threshold diminishes over time. Axonal delays might also be integrated, resulting in a time lag for outgoing neuron information to influence its synapses [69].

Synapses form the connections between neurons, each with pre-synaptic and post-synaptic constituents [33]. They possess weight values that could

be excitatory or inhibitory and have associated delay values, influencing the speed at which communications from the pre-synaptic neuron reach the post-synaptic one. Notably, learning mechanisms are often embedded within synapses, enabling the synaptic weight to change based on network activity. In this way, SNNs capture the essence of temporal dynamics, inherently facilitating event-driven or asynchronous operation, and harmonizing with the design of neuromorphic hardware [26].

However, the domain of neuromorphic computing doesn't just stop at replicating biological functionality. It seeks to incorporate machine learning paradigms, striving to adapt and optimize SNNs in a manner similar to traditional deep learning approaches [58]. This ambition, however, faces a critical challenge: the non-differentiable activation functions of spiking neurons [18]. Additionally, the temporal component of SNNs complicates the learning process. Hence, numerous strategies are being explored to adapt deep learning techniques for SNNs.

One such strategy includes employing surrogate gradients and smoothed activation functions to calculate error gradients and adjust weights across layers [1] [51]. There have been attempts to compute spike error gradients as well, and these efforts have displayed promising classification performance on the MNIST dataset [15]. Additionally, approaches such as backpropagation through time and real-time recurrent learning have been demonstrated on neuromorphic datasets [72].

An alternative approach to neuromorphic computing revolves around training a traditional deep neural network (DNN) and subsequently mapping it to an SNN for inference purposes. The motivation behind this is to leverage the established training mechanics of DNNs, yielding high-performance models while potentially reducing energy consumption. However, such approaches must be wary of the trade-offs between DNNs and SNNs and the limitations inherent in emerging neuromorphic hardware systems, like reduced synaptic weight precision and device variation.

While mapping strategies and deep learning adaptations offer intriguing possibilities for SNNs, they often don't fully exploit the temporal computational capabilities inherent in SNNs. This underutilization can limit the

potential of SNNs to mimic what traditional artificial neural networks can already achieve. Therefore, to realize the full potential of neuromorphic computing, it's crucial to explore, design, and implement strategies that leverage the unique properties of SNNs and bring us closer to emulating the richness of biological computation.

2 Neural Architecture Search overview

The design and implementation of neural network architectures can be a complex, labor-intensive, and expertise-reliant process. It involves the careful arrangement and tuning of multiple interconnected layers and components, each with its own parameters. A poorly designed network can result in subpar performance, while a well-crafted one can drastically improve the accuracy and efficiency of predictions. As a result, a significant part of the success of deep learning applications hinges on the architecture of the neural networks employed.

An approach to mitigate the need to carefully hand-craft the structure of the neural network is to use Neural Architecture Search (NAS). NAS is an automated approach that uses machine learning to design neural network architectures, effectively replacing the traditionally manual and heuristic-based process. By systematically exploring the space of possible architectures, NAS aims to discover network structures that yield the best performance for a given task.

The importance of NAS is multifold. First, NAS helps to reduce the reliance on expert knowledge in network design, making deep learning more accessible. Second, it can potentially uncover novel network structures that human designers might overlook. Third, NAS can tailor architectures to specific tasks, datasets, or resource constraints, leading to models that are both efficient and effective [6].

Furthermore, NAS can potentially yield architectures that outperform handcrafted ones. As demonstrated by various studies [40] [21], NAS has been able to design architectures that set new state-of-the-art performances on several benchmark tasks. In an era where the demand for machine learning solutions continues to grow across various sectors, the ability to automate the design of effective and efficient neural networks is invaluable. Hence, NAS plays a pivotal role in the continued advancement of deep learning applications.

Given the importance of NAS in the context of this work, in this chapter, we will be thoroughly exploring a variety of Neural Architecture Search

algorithms. Our aim is to identify and select the most appropriate algorithm for our specific task. By delving into the different algorithms available, we can better understand their strengths and weaknesses, enabling us to make an informed decision that aligns with our project objectives.

In the realm of Neural Architecture Search (NAS), researchers employ a variety of computational paradigms for navigating the expansive and discrete space of potential model architectures. One such approach is the utilization of evolutionary methods. By employing the principles of natural selection, a population of model architectures is gradually refined, aiming to yield the optimal solution [46]. Each model's weights are passed onto the subsequent generation, an inheritance concept inspired by evolutionary biology. This process allows the population to adapt over generations, improving overall accuracy and efficiency [28].

However, the utilization of SuperNetworks offers a promising alternative to the exhaustive computations of traditional evolutionary methods. A SuperNetwork is a complex neural network that encapsulates a search space whose elements consist of subnetworks. The entire collection of neural models can be defined as the subnetworks of a larger SuperNet, with weights shared among the common blocks of these subnetworks [21]. A supernetwork can be visualized as a vast Directed Acyclic Graph (DAG), within which different candidate neural networks are represented as subgraphs. The distinctive aspect of a supernetwork is that the weights are distributed across different sub-architectures that have overlapping edges, each representing a potential path within the larger supernetwork. The underlying concept is to streamline the process by training a singular, all-encompassing supernetwork that encapsulates multiple design possibilities, instead of independently generating and training thousands of networks.

This weight-sharing concept, first introduced as Efficient NAS (ENAS), was revolutionarily faster than previous NAS methods by more than 1000x, in terms of GPU hours, while achieving comparable accuracy. Using a Long Short Term Memory (LSTM) controller, the shared weights of the SuperNetwork are updated with the gradients produced by a randomly selected candidate subnetwork. This dual process allows for simultaneous optimization of

SuperNetwork weights and a subnetwork structure [21].

However, the SuperNetworks approach has limitations in terms of size, which may restrict its use to smaller datasets or as a search for a block of the target model. An alternate method proposes the use of an over-parameterized SuperNetwork, reducing memory consumption and optimizing the weights using gradient descent [10].

A variant of this method, AutoHAS [4], uses a single SuperNetwork and defines the architecture for every subnetwork as a linear combination of basis operations. This allows for the simultaneous search of architecture and hyper-parameters. The weight of the SuperNetwork becomes the union of weights for all basis operations in each layer.

In contrast, the Differentiable Architecture Search (DARTS) approach [40] allows for weight sharing by searching for a cell structure to construct the higher-level architecture. This approach introduces continuous relaxation of the cell representation, allowing the cell structure to be optimized using gradient descent. The resulting architectures were comparable to or better than ENAS in a similar timeframe.

Building upon DARTS, the Gradient-based search approach using Differentiable Architecture Sampling (GDAS) [19] alternates between normal and reduction cells. The final architecture is formed by stacking many copies of the discovered cells. However, DARTS and its variants suffer from a depth gap, as cells are optimized separately from the final configuration. To address this, Progressive DARTS (PDARTS) [62] increases the network depth progressively during the search process.

Another critical concern with DARTS is the large memory consumption when applied to deep networks for high-resolution image classification. Partially-Connected DARTS [59] addresses this issue by taking into account only random image channels when updating coefficients for operations, thereby reducing memory usage.

Finally, reinforcement learning (RL) provides another powerful paradigm for architecture search. RL-based NAS methods enable a more controlled search that can potentially decrease the amount of computation used [16] [73]. In RL NAS, an agent, or controller, samples an architecture (action)

and receives a validation accuracy (reward). While these methods face the bottleneck of training each sampled architecture from scratch, methods like those presented in [22] propose the definition of actions as transformations of the existing network, effectively preserving previously learned weights and thus significantly reducing computation cost.

Upon evaluating these methods, we chose DARTS for our study due to its ability to optimize cell structures using gradient descent and its continuous relaxation of cell representation. Although it does suffer from high memory consumption and a depth gap problem, DARTS presents a good balance between computation time and performance, making it a suitable choice for our investigation, as this project would not be bottlenecked by the model depth.

3 Automatic speech recognition

Given the context and the importance of ASR in this work, in this chapter, we will delve into various ASR architectures that have been developed. This analysis will serve as a foundational stepping stone, guiding us in designing an architecture that is well suited for our task. By understanding the underlying mechanisms and strengths of different architectures, a promising starting point for our model will be determined.

In the world of Automatic Speech Recognition (ASR), rapid advancements have been witnessed over the last decade, largely propelled by notable progress in deep neural network (DNN) architecture design, improvements in data augmentation techniques, and the exponential growth in the availability of high-quality training datasets [14, 2, 68, 23]. These advancements have contributed to significant reductions in ASR word-error-rate, further enhancing the reliability and accuracy of these systems.

Nonetheless, achieving state-of-the-art performance in ASR models continues to pose considerable challenges. The training process for these models is often computationally intensive, demanding thousands of GPU hours to reach satisfactory convergence [14, 39]. Compounded by the necessity for hyper-parameter optimizations, computational loads in ASR training are exponentially increased.

Despite these complexities at a system level, novel architectural design continues to play a pivotal role across a range of application domains, including ASR [11, 65], computer vision [13, 36], and natural language processing (NLP) [7, 3]. However, effective architecture design is often a highly challenging task. It frequently hinges on the combination of extensive experience, in-depth domain knowledge, and a record of empirical successes.

Over recent years, the deep learning community has started to witness a shift towards automated techniques for discovering neural network architectures, moving away from the more traditional, manually designed alternatives. Neural Architecture Search (NAS) algorithms have proven to be exceptionally successful in identifying state-of-the-art architectures across various computer vision tasks [57, 66, 34, 64, 45, 70].

However, many NAS algorithms suffer from considerable computational requirements, often necessitating a large number of architectural variants to be trained [73]. Furthermore, the reproducibility of NAS algorithms across different researchers can be challenging, often due to non-standard use of training settings, such as hyperparameters, and subtle variations in architecture search spaces [38, 24].

Recent efforts have been made to mitigate these challenges by releasing various benchmark datasets for the NAS research community [20, 50, 49, 48]. These datasets often provide a direct correlation between an architectural variant and its post-training performances, which can be leveraged by a NAS algorithm to expedite the search process.

NAS Benchmarks have been established to tackle the difficulties in reproducing NAS research. NAS-Bench-101 dataset [48] and NAS-Bench-201 dataset [20], NAS-Bench-1shot1 [67], and NAS-Bench-301 [49] offer large and diverse collections of image classification models. More recently, NASBench-NLP dataset [50] was introduced, featuring models with custom recurrent cells suited for language modeling tasks.

Recently, increasing interest has been observed in the application of NAS for speech-related tasks, such as keyword spotting [52, 31], speaker verification [5, 63], and acoustic scene classification [53]. NAS has also made inroads in the ASR domain [30, 12, 25, 37], contributing to the ongoing improvements in model performance.

For example, Chen et al. [12] used a methodology based on vanilla DARTS to optimize a CNN-based feature extractor, achieving improvements over a VGG-based extractor. In contrast, Kim et al. [25] applied evolution-based search to optimize a micro-cell within a transformer architecture, while He et al. [30] used differentiable search, using P-DARTS [12] as the base, to optimize a convolutional model. The work by Baruwa et al. [37] is most closely related to our work, where both TIMIT and LibriSpeech datasets were used for studying the effects of architectural changes on ASR models.

4 Neural Architecture Search algorithm

Our approach hinges on leveraging the Differentiable Architecture Search (DARTS) algorithm [40] — a gradient-based method renowned for its efficiency. To accommodate the peculiarities of neuromorphic hardware, we implemented a modified version of the NASLib library [**?**], introducing several critical changes. First, we redefined the search scope for the architecture, limiting it to operations executable on neuromorphic hardware: Linear layer, ReLU, Sigmoid, Tanh, Absolute, and Zero (indicating no connection). We also penned additional visualization code, and created a novel graph based on the efficient gradient-based architecture search approach delineated in a recent study, titled "DARTS-ASR" [12].

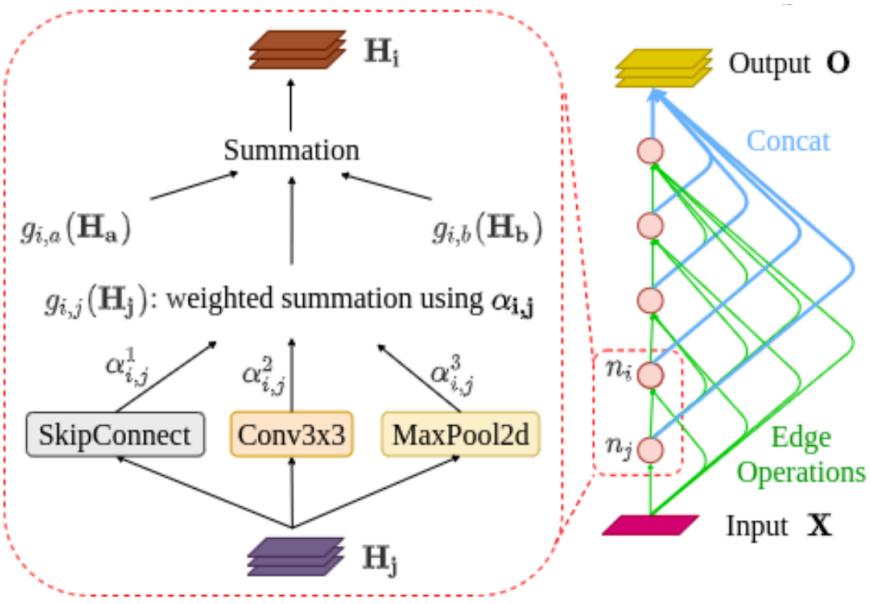

Figure 1: Differentiable ARchiTecture Search (DARTS) for ASR architecture

Building on this foundation, we ventured into uncharted territory, devising a method to process spiking datasets. In the domain of automatic speech recognition (ASR), the input feature is typically a segment of acoustic features, such as Mel-filterbanks [27]. Our architecture mirrors this concept, with the input feature, X, and latent features, $H_1, H_2, ..., H_K$, represented as nodes in a directed acyclic graph. The feature of each node is the summation of operations of all its preceding nodes, wherein each operation is the weighted sum of a set of transformations. For each node in the directed acyclic graph,

there are i directed input edges, where each edge transforms features (X or H_i) with some operation from the search space defined above. The final output of the architecture is the concatenation of all latent features.

In the Differentiable Architecture Search framework, architecture search is facilitated by learning a set of continuous variables $\{\alpha_{i,j}\}$. These variables, $\alpha_{i,j}$, play a critical role in determining the connections between nodes n_i and n_j, thereby implicitly controlling the overall architecture of the network.

The set $\{\alpha_{i,j}\}$ is trained jointly with the weights of the network through gradient descent optimization. This joint training allows the model to learn both its parameters and its structure simultaneously, significantly improving the efficiency of the architecture search.

A key aspect of DARTS is that if the weights $\alpha_{i,j}$ are sparse, it can be considered as selecting the transformations used to connect (or disconnect with zero operation) node n_i and n_j. This implies that $\alpha_{i,j}$ not only influences the architecture of the network but also determines the specific transformations employed between nodes.

The continuous relaxation of architecture representation by variables $\{\alpha_{i,j}\}$ allows the components of the model's transformations and its connections to be softly designed via gradient descent optimization. This effectively turns the traditionally discrete problem of architecture search into a continuous one that can be solved using standard optimization techniques. As such, DARTS provides an efficient and flexible approach for automated architecture search.

5 Experiments

5.1 Data and features

We tested this architecture on the Spiking Heidelberg Digits dataset [74], which comprises audio recordings of ten digits in English and German. To preserve the temporal structure of the data while enabling batching and shuffling, dataset and data loader classes were implemented.

The Heidelberg Digits (HD) dataset is an ideal candidate for testing our modified DARTS framework due to its unique structure and complexity. Audio has a temporal dimension, making it a natural choice for spike-based processing. However, unlike movie data, audio requires fewer input channels for a faithful representation, making the derived spiking datasets more computationally tractable. This dataset comprises roughly 10,000 high-quality recordings of spoken digits from zero to nine, in both English and German. It contains a diverse set of speakers, a total of 12, with ages ranging from 21 to 56 years old. The diversity in terms of language, digit, and speaker variety is an effective means of assessing the robustness and flexibility of the architecture search.

In preparing the HD dataset, an essential step involved the cutting and sequencing of the raw audio tracks. This preprocessing was performed by authors manually, supplemented by a blackbox-optimizer, ensuring accurate partitioning of the spoken digits. This intricate process was crucial in creating a dataset that closely mimics real-world scenarios, where spoken digits would typically be encountered in a sequence rather than isolation.

Lastly, the division of data into training and test sets was designed to test the generalization ability of the network. Two speakers were exclusively set aside for the test set, with the remainder of the test set supplemented by samples from speakers also present in the training set. This setup challenges the network's capacity to generalize across different speakers, further testing the efficacy of the architecture produced by our modified DARTS algorithm.

5.2 Implementation details

As this is yet to be ported to neuromorphic hardware, the training was performed on regular hardware. This NAS approach is built upon the Differentiable Architecture Search (DARTS) framework, so the key to this method is the dual optimizer scheme that includes an architecture optimizer and an operation optimizer. The architecture optimizer, built using Adam optimization algorithm [35], refines the continuous variables $\alpha_{i,j}$ that define the architecture of the network, including node connections and operation selection. In this optimizer, we set the learning rate to 0.02, with betas at (0.5, 0.999) and a weight decay of 0.001.

The operation optimizer, on the other hand, is constructed with the SGD algorithm, tasked to optimize the weights of the network. In this SGD optimizer, the learning rate, momentum, and weight decay are set at 0.025, 0.9, and 0.0003, respectively. The split between these two distinct optimization roles effectively decouples the evolution of the network architecture from the adaptation of the network weights, enabling a more efficient and stable learning process.

The architecture search runs over 10 epochs, with a batch size of 16 and a train-val split of 70:30. We also incorporate gradient clipping with a maximum value of 5 to prevent the gradients from exploding, which can destabilize the learning process. Our configuration allows the output weights to be saved for further analysis and reproducibility. Checkpointing ensures the work's progress can be recovered even in the case of unforeseen interruptions. The training was performed on one NVIDIA RTX A6000 GPU.

Our operation space includes six operations executable on neuromorphic hardware: Linear, ReLU, Sigmoid, Tanh, Absolute, and Zero. Notably, the size of our linear layers is set to (100, 100), fitting well within the hardware constraints, while other operations have no trainable parameters. The final operation for each edge is selected by a softmax operation over the $\alpha_{i,j}$ weights, effectively turning the differentiable architecture selection into a discrete one.

6 Results and conclusion

6.1 Results

In our study, we developed a Neural Architecture Search (NAS) algorithm with a particular focus on the constraints inherent in neuromorphic hardware. Specifically, we limited the search space to a selection of operations that are executable on this unique hardware, thus ensuring the produced architectures are not only efficient but also practical for implementation in a neuromorphic environment.

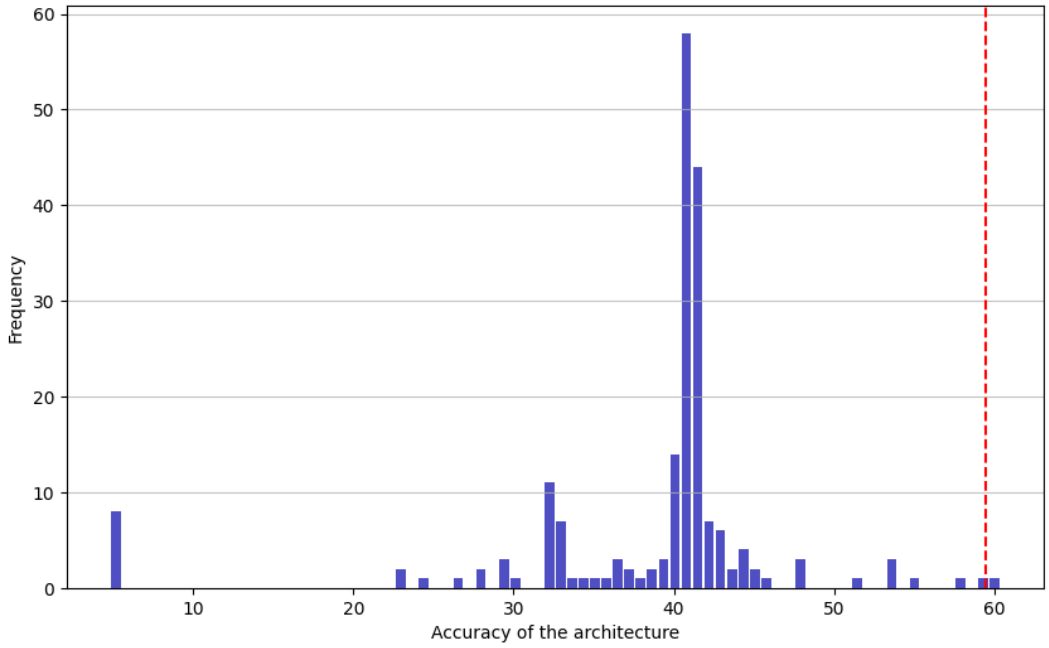

Figure 2: Distribution of random architectures sampled from the search space, vertical red line highlights the accuracy of the model found with DARTS

The implementation of our approach demonstrated promising results, identifying architectures that positioned themselves within the 90th percentile of a selection of 200 arbitrarily generated architectures, all coherently confined to the same search space (Fig. 2). This underscores the potential of our approach to pinpoint high-performing architecture designs, even within some of the constraints imposed by neuromorphic hardware.

Despite achieving compelling results, it should be noted that the identified architectures fell short of reaching the peak performance of state-of-the-art

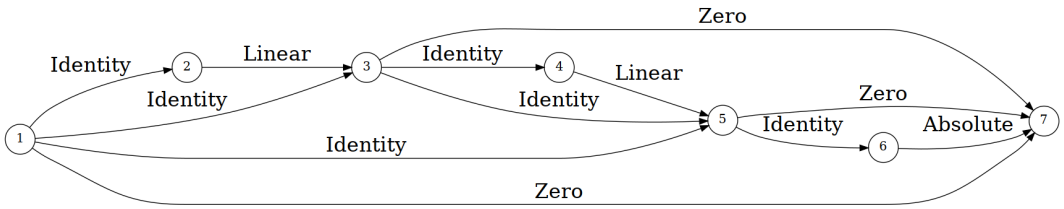

Figure 3: Found architecture with 0.58 accuracy

models. The most effective model identified through our approach (Fig. 3) yielded an accuracy of 0.58, as compared to the 0.92 achieved by the prevailing state-of-the-art model [8]. This divergence can be attributed to the limited trainable parameters within the search space defined for this project — 220 in comparison to the 404 parameters offered in the state-of-the-art model. However, if we compare the performance with the multi-layer perceptron model from the same article, we achieve comparable accuracy (0.58 vs 0.62) while having fewer trainable parameters (220 vs 404). Yet, in spite of this apparent shortcoming, the models presented commendable efficiency and demonstrated aptitude in handling tasks congruent with the neuromorphic framework.

A crucial observation from our study is the performance drop when transitioning from a continuous mixture of operations to a discrete architecture. This phenomenon is a well-known weakness inherent in the Differentiable Architecture Search (DARTS) algorithm, on which our method is built. The continuous relaxation of the architecture representation allows for efficient architecture search but translating these continuous representations to a practical, discrete architecture can often result in performance degradation. This reality highlights the challenges and trade-offs that we must grapple with in our pursuit of optimal neural network architectures, and it underlines the need for continued research and innovation in this field. Despite this limitation, our results confirm the overall viability and potential of our approach, suggesting exciting possibilities for the further refinement and application of NAS methods tailored for neuromorphic hardware. The code for this project is available at https://github.com/EIS-Hub/Spiking-NASLib

6.2 Conclusion

In conclusion, our study effectively demonstrates the potential of a tailored Neural Architecture Search (NAS) approach within the context of neuromorphic hardware. Although our identified architectures did not reach state-of-the-art performance levels, they nonetheless demonstrated their efficiency and suitability for tasks aligning with the neuromorphic paradigm. Despite being constrained by the limited set of trainable parameters, our approach successfully found superior architectures within the confined search space. This study affirms the exciting possibilities for further research and development in applying NAS methods specifically for neuromorphic hardware, paving the way for the potential to enhance the design and performance of spiking neural networks.

References

[1] Yin Shihui, Venkataramanaiah Shreyas K, Chen Gregory K, Krishnamurthy Ram, Cao Yu, Chakrabarti Chaitali, and Seo Jae-sun. Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations // 2017 IEEE Biomedical Circuits and Systems Conference (BioCAS) / IEEE. — 2017. — P. 1–5.

[2] Kim Kwangyoun, Lee Kyungmin, Gowda Dhananjaya, Park Junmo, Kim Sungsoo, Jin Sichen, Lee Young-Yoon, Yeo Jinsu, Kim Daehyun, Jung Seokyeong, et al. Attention based on-device streaming speech recognition with large speech corpus // 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU) / IEEE. — 2019. — P. 956–963.

[3] Vaswani Ashish, Shazeer Noam, Parmar Niki, Uszkoreit Jakob, Jones Llion, Gomez Aidan N, Kaiser Łukasz, and Polosukhin Illia. Attention is all you need // Advances in neural information processing systems. — 2017. — Vol. 30.

[4] Dong Xuanyi, Tan Mingxing, Yu Adams Wei, Peng Daiyi, Gabrys Bogdan, and Le Quoc V. AutoHAS: Efficient hyperparameter and architecture search // arXiv preprint arXiv:2006.03656. — 2020.

[5] Ding Shaojin, Chen Tianlong, Gong Xinyu, Zha Weiwei, and Wang Zhangyang. Autospeech: Neural architecture search for speaker recognition // arXiv preprint arXiv:2005.03215. — 2020.

[6] Baymurzina Dilyara, Golikov Eugene, and Burtsev Mikhail. A review of neural architecture search // Neurocomputing. — 2022. — Vol. 474. — P. 82–93.

[7] Devlin Jacob, Chang Ming-Wei, Lee Kenton, and Toutanova Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding // arXiv preprint arXiv:1810.04805. — 2018.

[8] Bittar Alexandre and Garner Philip N. A surrogate gradient spiking baseline for speech command recognition // Frontiers in Neuroscience. — 2022. — Vol. 16.

[9] Nandakumar SR, Kulkarni Shruti R, Babu Anakha V, and Rajendran Bipin. Building brain-inspired computing systems: Examining the role of nanoscale devices // IEEE Nanotechnology Magazine. — 2018. — Vol. 12, no. 3. — P. 19–35.

[10] Cai Han, Zhu Ligeng, and Han Song. Proxylessnas: Direct neural architecture search on target task and hardware // arXiv preprint arXiv:1812.00332. — 2018.

[11] Chiu Chung-Cheng and Raffel Colin. Monotonic chunkwise attention // arXiv preprint arXiv:1712.05382. — 2017.

[12] Chen Yi-Chen, Hsu Jui-Yang, Lee Cheng-Kuang, and Lee Hung-yi. DARTS-ASR: Differentiable architecture search for multilingual speech recognition and adaptation // arXiv preprint arXiv:2005.07029. — 2020.

[13] He Kaiming, Zhang Xiangyu, Ren Shaoqing, and Sun Jian. Deep residual learning for image recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 770–778.

[14] Amodei Dario, Ananthanarayanan Sundaram, Anubhai Rishita, Bai Jingliang, Battenberg Eric, Case Carl, Casper Jared, Catanzaro Bryan, Cheng Qiang, Chen Guoliang, et al. Deep speech 2: End-to-end speech recognition in english and mandarin // International conference on machine learning / PMLR. — 2016. — P. 173–182.

[15] Deng Li. The mnist database of handwritten digit images for machine learning research [best of the web] // IEEE signal processing magazine. — 2012. — Vol. 29, no. 6. — P. 141–142.

[16] Baker Bowen, Gupta Otkrist, Naik Nikhil, and Raskar Ramesh. Designing neural network architectures using reinforcement learning // arXiv preprint arXiv:1611.02167. — 2016.

[17] Islam Raisul, Li Haitong, Chen Pai-Yu, Wan Weier, Chen Hong-Yu, Gao Bin, Wu Huaqiang, Yu Shimeng, Saraswat Krishna, and Wong HS Philip. Device and materials requirements for neuromorphic computing // Journal of Physics D: Applied Physics. — 2019. — Vol. 52, no. 11. — P. 113001.

[18] Li Yuhang, Guo Yufei, Zhang Shanghang, Deng Shikuang, Hai Yongqing, and Gu Shi. Differentiable spike: Rethinking gradient-descent for training spiking neural networks // Advances in Neural Information Processing Systems. — 2021. — Vol. 34. — P. 23426–23439.

[19] Dong Xuanyi and Yang Yi. Searching for a robust neural architecture in four gpu hours // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2019. — P. 1761–1770.

[20] Dong Xuanyi and Yang Yi. Nas-bench-201: Extending the scope of reproducible neural architecture search // arXiv preprint arXiv:2001.00326. — 2020.

[21] Pham Hieu, Guan Melody Y., Zoph Barret, Le Quoc V., and Dean Jeff. Efficient Neural Architecture Search via Parameter Sharing. — 2018. — 1802.03268.

[22] Cai Han, Chen Tianyao, Zhang Weinan, Yu Yong, and Wang Jun. Efficient architecture search by network transformation // Proceedings of the AAAI Conference on Artificial Intelligence. — 2018. — Vol. 32.

[23] Synnaeve Gabriel, Xu Qiantong, Kahn Jacob, Likhomanenko Tatiana, Grave Edouard, Pratap Vineel, Sriram Anuroop, Liptchinsky Vitaliy, and Collobert Ronan. End-to-end asr: from supervised to semi-supervised learning with modern architectures // arXiv preprint arXiv:1911.08460. — 2019.

[24] Yu Kaicheng, Sciuto Christian, Jaggi Martin, Musat Claudiu, and Salzmann Mathieu. Evaluating the search phase of neural architecture search // arXiv preprint arXiv:1902.08142. — 2019.

[25] Kim Jihwan, Wang Jisung, Kim Sangki, and Lee Yeha. Evolved Speech-Transformer: Applying Neural Architecture Search to End-to-End Automatic Speech Recognition. // INTERSPEECH. — 2020. — P. 1788–1792.

[26] Rathi Nitin, Chakraborty Indranil, Kosta Adarsh, Sengupta Abhronil, Ankit Aayush, Panda Priyadarshini, and Roy Kaushik. Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware // ACM Computing Surveys. — 2023. — Vol. 55, no. 12. — P. 1–49.

[27] Fayek Haytham M. Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between. — 2016. — Access mode: `https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html`.

[28] Floreano Dario, Dürr Peter, and Mattiussi Claudio. Neuroevolution: from architectures to learning // Evolutionary intelligence. — 2008. — Vol. 1. — P. 47–62.

[29] Sze Vivienne, Chen Yu-Hsin, Emer Joel, Suleiman Amr, and Zhang Zhengdong. Hardware for machine learning: Challenges and opportunities // 2017 IEEE Custom Integrated Circuits Conference (CICC) / IEEE. — 2017. — P. 1–8.

[30] He Liqiang, Su Dan, and Yu Dong. Learned transferable architectures can surpass hand-designed architectures for large scale speech recognition // ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) / IEEE. — 2021. — P. 6788–6792.

[31] Mazzawi Hanna, Gonzalvo Xavi, Kracun Aleks, Sridhar Prashant, Subrahmanya Niranjan, Lopez-Moreno Ignacio, Park Hyun-Jin, and Violette Patrick. Improving Keyword Spotting and Language Identification via Neural Architecture Search at Scale. // Interspeech. — 2019. — P. 1278–1282.

[32] Irizarry-Valle Yilda and Parker Alice Cline. An astrocyte neuromorphic circuit that influences neuronal phase synchrony // IEEE transactions on biomedical circuits and systems. — 2015. — Vol. 9, no. 2. — P. 175–187.

[33] Jin Yingyezhe, Zhang Wenrui, and Li Peng. Hybrid macro/micro level backpropagation for training deep spiking neural networks // Advances in neural information processing systems. — 2018. — Vol. 31.

[34] Lee Royson, Dudziak Łukasz, Abdelfattah Mohamed, Venieris Stylianos I, Kim Hyeji, Wen Hongkai, and Lane Nicholas D. Journey towards tiny perceptual super-resolution // Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI / Springer. — 2020. — P. 85–102.

[35] Kingma Diederik P and Ba Jimmy. Adam: A method for stochastic optimization // arXiv preprint arXiv:1412.6980. — 2014.

[36] Krizhevsky Alex, Sutskever Ilya, and Hinton Geoffrey E. Imagenet classification with deep convolutional neural networks // Communications of the ACM. — 2017. — Vol. 60, no. 6. — P. 84–90.

[37] Baruwa Ahmed, Abisiga Mojeed, Gbadegesin Ibrahim, and Fakunle Afeez. Leveraging end-to-end speech recognition with neural architecture search // arXiv preprint arXiv:1912.05946. — 2019.

[38] Li Liam and Talwalkar Ameet. Random search and reproducibility for neural architecture search // Uncertainty in artificial intelligence / PMLR. — 2020. — P. 367–377.

[39] Kahn Jacob, Riviere Morgane, Zheng Weiyi, Kharitonov Evgeny, Xu Qiantong, Mazaré Pierre-Emmanuel, Karadayi Julien, Liptchinsky Vitaliy, Collobert Ronan, Fuegen Christian, et al. Libri-light: A benchmark for asr with limited or no supervision // ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) / IEEE. — 2020. — P. 7669–7673.

[40] Liu Hanxiao, Simonyan Karen, and Yang Yiming. Darts: Differentiable architecture search // arXiv preprint arXiv:1806.09055. — 2018.

[41] Davies Mike, Srinivasa Narayan, Lin Tsung-Han, Chinya Gautham, Cao Yongqiang, Choday Sri Harsha, Dimou Georgios, Joshi Prasad, Imam Nabil, Jain Shweta, et al. Loihi: A neuromorphic manycore processor with on-chip learning // Ieee Micro. — 2018. — Vol. 38, no. 1. — P. 82–99.

[42] Mayr Christian, Hoeppner Sebastian, and Furber Steve. Spinnaker 2: A 10 million core processor system for brain simulation and machine learning // arXiv preprint arXiv:1911.02385. — 2019.

[43] Mead Carver. Neuromorphic electronic systems // Proceedings of the IEEE. — 1990. — Vol. 78, no. 10. — P. 1629–1636.

[44] Najem Joseph S, Taylor Graham J, Weiss Ryan J, Hasan Md Sakib, Rose Garrett, Schuman Catherine D, Belianinov Alex, Collier C Patrick, and Sarles Stephen A. Memristive ion channel-doped biomembranes as synaptic mimics // ACS nano. — 2018. — Vol. 12, no. 5. — P. 4702–4711.

[45] Tan Mingxing, Chen Bo, Pang Ruoming, Vasudevan Vijay, Sandler Mark, Howard Andrew, and Le Quoc V. Mnasnet: Platform-aware neural architecture search for mobile // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. — 2019. — P. 2820–2828.

[46] Montana David J, Davis Lawrence, et al. Training feedforward neural networks using genetic algorithms. // IJCAI. — 1989. — Vol. 89. — P. 762–767.

[47] Mostafa Hesham, Müller Lorenz K, and Indiveri Giacomo. An event-based architecture for solving constraint satisfaction problems // Nature communications. — 2015. — Vol. 6, no. 1. — P. 8941.

[48] Ying Chris, Klein Aaron, Christiansen Eric, Real Esteban, Murphy Kevin, and Hutter Frank. Nas-bench-101: Towards reproducible

neural architecture search // International Conference on Machine Learning / PMLR. — 2019. — P. 7105–7114.

[49] Siems Julien, Zimmer Lucas, Zela Arber, Lukasik Jovita, Keuper Margret, and Hutter Frank. Nas-bench-301 and the case for surrogate benchmarks for neural architecture search // arXiv preprint arXiv:2008.09777. — 2020. — Vol. 11.

[50] Klyuchnikov Nikita, Trofimov Ilya, Artemova Ekaterina, Salnikov Mikhail, Fedorov Maxim, Filippov Alexander, and Burnaev Evgeny. Nas-bench-nlp: neural architecture search benchmark for natural language processing // IEEE Access. — 2022. — Vol. 10. — P. 45736–45747.

[51] Neftci Emre O, Mostafa Hesham, and Zenke Friedemann. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks // IEEE Signal Processing Magazine. — 2019. — Vol. 36, no. 6. — P. 51–63.

[52] Mo Tong, Yu Yakun, Salameh Mohammad, Niu Di, and Jui Shangling. Neural architecture search for keyword spotting // arXiv preprint arXiv:2009.00165. — 2020.

[53] Li Jixiang, Liang Chuming, Zhang Bo, Wang Zhao, Xiang Fei, and Chu Xiangxiang. Neural architecture search on acoustic scene classification // arXiv preprint arXiv:1912.12825. — 2019.

[54] Neuromorphic Computing, Architectures, Models, and Applications. A Beyond-CMOS Approach to Future Computing, June 29-July 1, 2016, Oak Ridge, TN : Rep. / USDOE Office of Science (SC)(United States). Advanced Scientific Computing ... ; executor: Potok Thomas, Schuman Catherine, Patton Robert, Hylton Todd, Li Hai, and Pino Robinson : 2016.

[55] Sandamirskaya Yulia, Kaboli Mohsen, Conradt Jorg, and Celikel Tansu. Neuromorphic computing hardware and neural architectures for robotics // Science Robotics. — 2022. — Vol. 7, no. 67. — P. eabl8419.

[56] Schuman Catherine D, Plank James S, Bruer Grant, and Anantharaj Jeremy. Non-traditional input encoding schemes for spiking neuromorphic systems // 2019 International Joint Conference on Neural Networks (IJCNN) / IEEE. — 2019. — P. 1–10.

[57] Cai Han, Gan Chuang, Wang Tianzhe, Zhang Zhekai, and Han Song. Once-for-all: Train one network and specialize it for efficient deployment // arXiv preprint arXiv:1908.09791. — 2019.

[58] Schuman Catherine D, Kulkarni Shruti R, Parsa Maryam, Mitchell J Parker, Date Prasanna, and Kay Bill. Opportunities for neuromorphic computing algorithms and applications // Nature Computational Science. — 2022. — Vol. 2, no. 1. — P. 10–19.

[59] Xu Yuhui, Xie Lingxi, Zhang Xiaopeng, Chen Xin, Qi Guo-Jun, Tian Qi, and Xiong Hongkai. Pc-darts: Partial channel connections for memory-efficient architecture search // arXiv preprint arXiv:1907.05737. — 2019.

[60] Pfeiffer Michael and Pfeil Thomas. Deep learning with spiking neurons: Opportunities and challenges // Frontiers in neuroscience. — 2018. — Vol. 12. — P. 774.

[61] Polykretis Ioannis, Tang Guangzhi, and Michmizos Konstantinos P. An astrocyte-modulated neuromorphic central pattern generator for hexapod robot locomotion on intel's loihi // International Conference on Neuromorphic Systems 2020. — 2020. — P. 1–9.

[62] Chen Xin, Xie Lingxi, Wu Jun, and Tian Qi. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation // Proceedings of the IEEE/CVF international conference on computer vision. — 2019. — P. 1294–1303.

[63] Qu Xiaoyang, Wang Jianzong, and Xiao Jing. Evolutionary algorithm enhanced neural architecture search for text-independent speaker verification // arXiv preprint arXiv:2008.05695. — 2020.

[64] Real Esteban, Aggarwal Alok, Huang Yanping, and Le Quoc V. Regularized evolution for image classifier architecture search // Proceedings of the aaai conference on artificial intelligence. — 2019. — Vol. 33. — P. 4780–4789.

[65] Pratap Vineel, Xu Qiantong, Kahn Jacob, Avidov Gilad, Likhomanenko Tatiana, Hannun Awni, Liptchinsky Vitaliy, Synnaeve Gabriel, and Collobert Ronan. Scaling up online speech recognition using convnets // arXiv preprint arXiv:2001.09727. — 2020.

[66] Howard Andrew, Sandler Mark, Chu Grace, Chen Liang-Chieh, Chen Bo, Tan Mingxing, Wang Weijun, Zhu Yukun, Pang Ruoming, Vasudevan Vijay, et al. Searching for mobilenetv3 // Proceedings of the IEEE/CVF international conference on computer vision. — 2019. — P. 1314–1324.

[67] Siems Julien, Zela Arber, and Hutter Frank. Towards Benchmarking and Dissecting One-shot Neural Architecture Search // Cell. — Vol. 1. — P. 1.

[68] Park Daniel S, Chan William, Zhang Yu, Chiu Chung-Cheng, Zoph Barret, Cubuk Ekin D, and Le Quoc V. Specaugment: A simple data augmentation method for automatic speech recognition // arXiv preprint arXiv:1904.08779. — 2019.

[69] Sun Pengfei, Zhu Longwei, and Botteldooren Dick. Axonal delay as a short-term memory for feed forward deep spiking neural networks // ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) / IEEE. — 2022. — P. 8932–8936.

[70] Tan Mingxing and Le Quoc. Efficientnet: Rethinking model scaling for convolutional neural networks // International conference on machine learning / PMLR. — 2019. — P. 6105–6114.

[71] Pei Jing, Deng Lei, Song Sen, Zhao Mingguo, Zhang Youhui, Wu Shuang, Wang Guanrui, Zou Zhe, Wu Zhenzhi, He Wei, et al. Towards artificial general intelligence with hybrid Tianjic chip architecture // Nature. — 2019. — Vol. 572, no. 7767. — P. 106–111.

[72] Zenke Friedemann and Neftci Emre O. Brain-inspired learning on neuromorphic substrates // Proceedings of the IEEE. — 2021. — Vol. 109, no. 5. — P. 935–950.

[73] Zoph Barret and Le Quoc V. Neural architecture search with reinforcement learning // arXiv preprint arXiv:1611.01578. — 2016.

[74] Cramer Benjamin, Stradmann Yannik, Schemmel Johannes, and Zenke Friedemann. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks // IEEE Transactions on Neural Networks and Learning Systems. — 2020. — Vol. 33, no. 7. — P. 2744–2757.

[75] Amir Arnon, Taba Brian, Berg David, Melano Timothy, McKinstry Jeffrey, Di Nolfo Carmelo, Nayak Tapan, Andreopoulos Alexander, Garreau Guillaume, Mendoza Marcela, et al. A low power, fully event-based gesture recognition system // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2017. — P. 7243–7252.

[76] Merolla Paul A, Arthur John V, Alvarez-Icaza Rodrigo, Cassidy Andrew S, Sawada Jun, Akopyan Filipp, Jackson Bryan L, Imam Nabil, Guo Chen, Nakamura Yutaka, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface // Science. — 2014. — Vol. 345, no. 6197. — P. 668–673.

[77] Furber Steve B, Galluppi Francesco, Temple Steve, and Plana Luis A. The spinnaker project // Proceedings of the IEEE. — 2014. — Vol. 102, no. 5. — P. 652–665.

[78] Schemmel Johannes, Brüderle Daniel, Grübl Andreas, Hock Matthias, Meier Karlheinz, and Millner Sebastian. A wafer-scale neuromorphic hardware system for large-scale neural modeling // 2010 ieee international symposium on circuits and systems (iscas) / IEEE. — 2010. — P. 1947–1950.