Interspeech 2024
1-5 September 2024, Kos, Greece

# Efficient CNNs with Quaternion Transformations and Pruning for Audio Tagging

*Aryan Chaudhary*[1,+], *Arshdeep Singh*[2,+], *Vinayak Abrol*[1], *Mark D. Plumbley*[2]

[1]CSE Department and Infosys Centre for AI, IIIT Delhi, India
[2]Centre for Vision, Speech and Signal Processing, University of Surrey, UK

aryan22019@iiitd.ac.in, arshdeep.singh@surrey.ac.uk, abrol@iiitd.ac.in, m.plumbley@surrey.ac.uk

## Abstract

This paper presents a novel approach to make convolutional neural networks (CNNs) efficient by reducing their computational cost and memory footprint. Even though large-scale CNNs show state-of-the-art performance in many tasks, high computational costs and the requirement of a large memory footprint make them resource-hungry. Therefore, deploying large-scale CNNs on resource-constrained devices poses significant challenges. To address this challenge, we propose to use quaternion CNNs, where quaternion algebra enables the memory footprint to be reduced. Furthermore, we investigate methods to reduce the memory footprint and computational cost further through pruning the quaternion CNNs. Experimental evaluation of the audio tagging task involving the classification of 527 audio events from AudioSet shows that the quaternion algebra and pruning reduce memory footprint by 90% and computational cost by 70% compared to the original CNN model while maintaining similar performance.

**Index Terms**: Audio tagging, CNNs, Quaternion neural networks, Pruning, Compression.

## 1. Introduction

Everyday acoustic environment contains a diverse array of sounds, from traffic and construction to subtle noises like keys jangling, music playing or conversations [1]. This rich soundscape holds vast information useful for surveillance, healthcare, and improving environments in workplaces or cities [2]. Techniques for recognizing these sound activities, known as audio tagging, involve capturing environmental sounds with microphones and identifying different activities [3].

In recent years, commercial interest in artificial intelligence (AI) integration into edge devices for personalized experiences and privacy solutions [4] has reshaped user interactions, emphasizing the demand for low-latency, memory-efficient AI models [5]. Recent advancements in deep learning (DL), especially convolutional neural networks (CNNs), have shown remarkable performance in tasks like audio tagging. However, large-scale CNN models present challenges due to their resource-intensive nature, requiring substantial computational power and memory resources. For example, cutting-edge CNN frameworks like pre-trained audio neural networks (PANNs) [6] have 81M parameters, occupying 312MB of memory & require over 20G computations to process 10 seconds of audio. The high computational complexity results in high inference latency and energy inefficiency, especially considering the large memory footprint of CNNs. Consequently, deploying such large-scale CNNs on mobile phones or IoT devices presents substantial challenges due to their limited computational/memory capabilities.

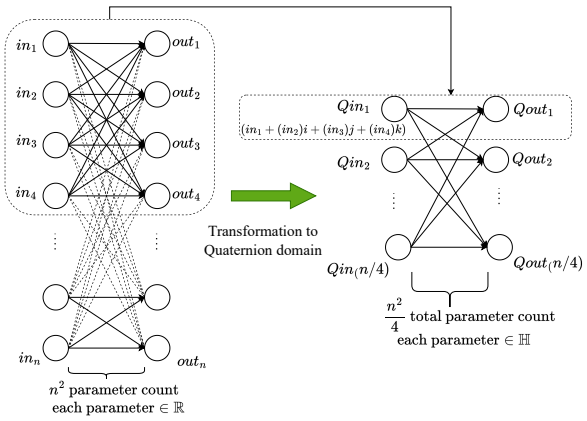


Figure 1: *Transformation of a conventional neural network in quaternion domain.*

The most popular method for enhancing CNN architecture efficiency is model compression [5]. Typically, model compression involves simplifying or removing elements of CNN models by filter pruning methods, where few learned filters contributing least to performance are pruned away from CNNs [7]. An alternative method is to use knowledge distillation (KD) from large-scale CNNs to design low-complexity, smaller memory footprint student models. However, selecting an optimal student network involves intricate engineering, network size tuning and demands substantial amounts of unlabeled data and entails significant training costs [8, 9].

In this work, we explore Quaternion convolutional neural networks (QCNNs) [10] in combination with architectural refinement through filter pruning [11] as an alternative for effective acoustic modelling for the audio-tagging task. While convolutional operation in CNNs operates with real-valued data and convolutional filters, QCNNs operate with quaternion data and quaternion filters, which are hypercomplex numbers consisting of a real part and three imaginary parts [12]. Unlike multi-channel models that ignore internal correlation among channels, QCNNs can embed various facets of input features along with spatial transformations and can better learn the complex relationships within them [13]. QCNNs employ the Hamilton product instead of the standard dot product, which results in generalization across different orientations, scales, and translations more effectively with very few parameters that are shared across channels [14]. An illustration to transform a conventional neural network to quaternion domain that results in re-

[+]Both authors contributed equally.


10.21437/Interspeech.2024-1331

ducing total parameter count is shown in Figure 1. More details about the quaternion transformation is explained in Section 3.

QCNN models have recently gained traction in speech/audio tasks such as speech recognition [15], event detection [16] and keyword-spotting [17]. However, a QCNN layer requires more computations than a traditional CNN layer. To mitigate the computational complexity inherent in QCNNs, we implement a strategy of pruning quaternion filters that minimally impact the network's output performance. The filter pruning process involves the targeted removal of entire quaternion filters and their associated quaternion output. Such a methodical reduction not only decreases the computational complexity but also leads to a significant decrease in the number of parameters within the QCNNs. As a result, we achieve a streamlined version of QCNNs, termed Efficient QCNNs (E-QCNNs), which can yield similar performance while operating with reduced computational overhead and enhanced parameter efficiency compared to original CNNs. Further, we demonstrate that pruning large-scale PANNs alone achieves suboptimal performance compared to the proposed E-QCNN model that leverages quaternion algebra and pruning to reduce computations and memory storage. To this aim, we used the popular large-scale AudioSet benchmark [3] to validate our approach using CNN14, one of the best-performing PANNs models [6]. The exact model architecture (with hyper-parameters), training/testing recipe and pre-trained model weights are accessible online*. The key advantages and major contributions of this paper can be summarized as follows:

- We propose a simple yet effective framework to reduce parameter count in CNNs by transforming CNNs to QCNNs by leveraging quaternion algebra. Further, we apply pruning to obtain efficient QCNNs (E-QCNNs) to reduce the computational complexity of QCNNs and the parameter count.
- Our framework reduces computations per inference by 70% with 90% fewer parameters with a marginal reduction in performance compared to that of the original CNN.
- Empirically, we demonstrate that E-QCNN obtained via leveraging both quaternion algebra and pruning is advantageous in terms of improved performance, fewer computations, and smaller parameter counts compared to obtaining efficient CNNs by applying pruning alone.

## 2. Related Works

The field of audio tagging has seen significant advancements with the introduction of AudioSet dataset [3], a comprehensive collection of over 2 million audio clips labelled across 527 sound classes. Popular deep learning models for audio tagging include large-scale convolutional neural networks (CNNs), recurrent neural networks (RNNs), and their variants, which have demonstrated significant improvements over traditional machine learning methods [6]. Audio-visual learning with contrastive audio-visual masked autoencoder methods has further helped improve the tagging accuracy of these models as they enable a model's joint learning in audio and visual space [18]. Large-scale transformer models and capsule networks have also shown promising results for this task both in supervised and unsupervised settings. The current state-of-the-art (SOTA) for audio tagging include multimodal transformer architecture based autoencoder [18] and model agnostic methods based on pre-training, balanced sampling, data augmentation and label enhancement [19]. Although current SOTA models are accurate for audio tagging, due to high computational complexity and large memory requirements, such well performing models are difficult to deploy on edge devices as optimizing such complex architectures for edge devices is not feasible.

Recently, there has been a focus on reducing the computational and memory requirements of CNNs through pruning techniques. Pruning methods involve eliminating a subset of parameters, such as weights or convolutional filters, from CNNs that contribute minimally to their overall performance [20, 21]. For example, Li et al. [11] found eliminating 64% parameters that yield 34% computations does not affect the performance much. Pruning can be performed either by eliminating whole convolutional filters [22] or by eliminating individual weights [23]. In the former case, the resultant network is a structured pruned model, where cross-platform inference is supported in off-the-shelf libraries. However, the later case results in an unstructured sparse pruned model that requires specialised software or hardware for speed-up [24]. Therefore, filter pruning methods are advantageous compared to weight pruning methods. Most filter pruning methods are active, where the importance of the filters is measured using a training dataset. Such active filter pruning methods either involve joint optimization of network parameters while computing filter importance [25, 26] or use filter outputs to measure importance [27, 28]. In contrast, passive pruning methods are data-independent as they directly compute importance using trained filter weights [11, 29].

## 3. Efficient Quaternion CNN (E-QCNN)

### 3.1. Quaternion Preliminaries

Quaternions are hypercomplex numbers extending the complex domain [30] using a real and three imaginary components as

$$Q = r + x\mathbf{i} + y\mathbf{j} + z\mathbf{k};$$

$$r, x, y, z \in \mathbb{R}; \quad \mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1.$$

In quaternion models, the standard dot product is replaced with the Hamilton product. Quaternions facilitate swift operations analogous to those performed with matrices, thanks to the existence of a lossless one-to-one mapping for both their addition and multiplication processes while maintaining the structural properties of these operations [31]. For example, the Hamiltonian product ($\otimes$) between two quaternions $Q_1 = r_1 + x_1\mathbf{i} + y_1\mathbf{j} + z_1\mathbf{k}$, and $Q_2 = r_2 + x_2\mathbf{i} + y_2\mathbf{j} + z_2\mathbf{k}$ can be efficiently expressed in terms of the products of the basis elements as

$$\begin{aligned} Q_1 \otimes Q_2 =& (r_1 r_2 - x_1 x_2 - y_1 y_2 - z_1 z_2) + \\ & (r_1 x_2 + x_1 r_2 + y_1 z_2 - z_1 y_2)\boldsymbol{i} + \\ & (r_1 y_2 - x_1 z_2 + y_1 r_2 + z_1 x_2)\boldsymbol{j} + \\ & (r_1 z_2 + x_1 y_2 - y_1 x_2 + z_1 r_2)\boldsymbol{k}. \end{aligned}$$

Quaternion operations can lead to $4\times$ parameter savings in a deep neural network (see Figure 1 for an illustration). A dense neural network layer with $n$ inputs and outputs results in a total $n^2$ number of parameters. In contrast, the quaternion domain encapsulates $n$ real input units into $\frac{n}{4}$ units in the hypercomplex domain and represents each parameter in the quaternion neural network in the hypercomplex domain. Thus, quaternion transformation yields ($\frac{n}{4} \times \frac{n}{4} \times 4 = \frac{n^2}{4}$) parameter, reducing the total number of parameters by 4 times. Further, note that nonlinearity can be incorporated using quaternion split activation $\sigma$ defined in terms of a standard activation function $\bar{\sigma}$ as

$$\sigma(Q) = \bar{\sigma}(r) + \bar{\sigma}(x)\mathbf{i} + \bar{\sigma}(y)\mathbf{j} + \bar{\sigma}(z)\mathbf{k}.$$

*https://github.com/Cross-Caps/QPANN

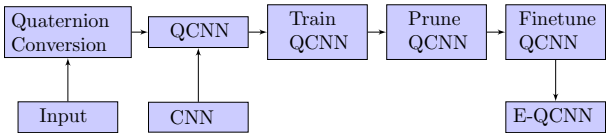


Figure 2: *Proposed framework to obtain an E-QCNN model.*

### 3.2. Building E-QCNN model for Audio Tagging

The overall framework for obtaining an E-QCNN model is illustrated in Figure 2, which comprises three stages, namely 1) representing input data in the quaternion domain and transforming the original CNN architecture into QCNN; 2) training the QCNN to achieve comparable performance; and 3) fine-tuning the pretrained QCNN model after pruning irrelevant filters.

#### *3.2.1. Quaternion model conversion*

Designing a quaternion variant of a conventional CNN model requires replacing 1) the regular matrix-vector product with the Hamiltonian product and 2) the activation function with a split activation function. In addition, note that we split the channels into four groups, each representing the real components and the $\mathbf{i}, \mathbf{j}, \mathbf{k}$ imaginary components. This is similar to group convolution, with the constraint that the number of channels should be divisible by 4. However, unlike multiple individual kernels per group, quaternion kernels are shared, i.e., they interact with each group to produce the final output. Assume $\gamma^l_{uv}$ and $O^l_{uv}$ be the quaternion output and the pre-activation quaternion output at layer $l$ and output indexes $(u, v)$. The quaternion convolution operation is mathematically expressed as [32]:

$$\gamma^l_{uv} = \sigma(O^l_{uv}), \quad O^l_{uv} = \sum_{u'=1}^{C-1} \sum_{v'=1}^{C-1} w^{l-1}_{u'v'} \otimes \gamma^{l-1}_{(u+u')(v+v')}.$$

where $w_{u'v'}$ is the quaternion-valued weight filter for map of size $C \times C$ and $\sigma$ is a quaternion split activation.

#### *3.2.2. Pruning quaternion filters*

Parameter saving in a QCNN doesn't result in lower computational complexity in addition/multiplication operations due to shared kernel interactions across various channel groups. To address this, we propose to use filter pruning as a post-processing step to reduce the computational complexity inherent in QCNNs with minimal impact on the network's performance. In this work, we extend the passive filter pruning method from Li et al. [11] to QCNNs due to its simplicity, where the pruning is achieved in a layer-by-layer manner. Given a set of QCNN filters, we prune away $p$ least important quaternion filters along real, $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ channels independently. The importance of the filters along each channel is measured using their sum of absolute coefficients or $\ell_1$-norm. As demonstrated by Li et al. [11], we observed that quaternion filters exhibit similar behaviour, i.e., filters with smaller $\ell_1$-norm produce output with weak activation compared to other quaternion filters. Hence, such filters are less significant and eliminating them has a negligible effect on the performance in practice. After eliminating $p$ least significant quaternion filters along each channel in $l^{th}$ layer, we apply the same procedure for other layers as well. After applying pruning across various intermediate layers, we obtain a pruned network that has a reduced number of parameters and computations compared to that of vanilla QCNN. In the end, we perform a final fine-tuning step on the E-QCNN to regain most of the performance lost due to the pruning procedure.

## 4. Experiment Setup

This section provides a system description, experimental protocol, and the dataset used in the experimental study.

### 4.1. Dataset

In this work, we have considered the AudioSet dataset [3] that contains 2 million 10-second-long utterances from 527 audio event classes. We use the standard train/validation/evaluation set and report the average mAPs over 3 trials on the evaluation set. In all experiments, the standard deviation in mAPs across trials is found to be approximately $10^{-3}$.

### 4.2. Evaluation metric

We report a model's performance using the mean average precision (mAPs) metric, a popular robust metric for imbalanced datasets like AudioSet. In addition, we compare computational efficiency using parameter count and multiply-accumulate operations (MACs) of various CNN models.

### 4.3. Model architectures

In this work, we experimented with CNN14, a well-performing model from PANNs [6] as our baseline model with 0.431 mAPs on the AudioSet evaluation set. This model consists of a total of 12 convolutional layers, among which the last six layers contribute more than 90% of the total 81M parameters (with 21G MACs). CNN14 is pre-trained on AudioSet using LogMel spectrograms of size $(1000 \times 64)$ from 10s long audio inputs sampled at 32KHz with a window size of 1024 samples and a hop size of 320 samples. The quaternion versions of CNN14, i.e., QCNN & E-QCNN are trained with LogMel spectrograms-based acoustic quaternions as inputs. To have an extensive comparison, we also have considered a comparison with various other CNN architectures (with different parameter budgets); namely plain CNNs (CNN6, CNN10) [6], residual networks (ResNet22 [6], ResNet54 [6], DeepRes [33]), efficient networks (MobileNetV1 [6], MobileNetV2 [6], MobileNetV3 [34, 35], EfficientNet-B2 [19, 36]) and E-PANNs a pruned only variant of CNN14 [29].

### 4.4. Acoustic quaternions representations

In this work, we employ Logmel-spectrogram based acoustic quaternion $Q(f, t)$ defined for each frequency $f$ and time frame $t$. This quaternion is composed of the energy $\psi(f, t)$ within the Mel-filter band at frequency $f$, along with its first-order (velocity), second-order (acceleration), and third-order (jerk) time derivatives, as described in [17]:

$$Q(f,t) = \psi(f,t) + \frac{\partial \psi(f,t)}{\partial t}\mathbf{i} + \frac{\partial^2 \psi(f,t)}{\partial^2 t}\mathbf{j} + \frac{\partial^3 \psi(f,t)}{\partial^3 t}\mathbf{k}$$

This allows QCNN to capture the spatial relationships across different temporal perspectives of the same frequency.

### 4.5. QCNN model training and pruning

The training or fine-tuning procedure for QCNN is similar to CNN14 as described in [6]. Each QCNN model is trained until the performance converges to CNN14 performance, or the QCNN model is trained for 500k epochs with cross-entropy

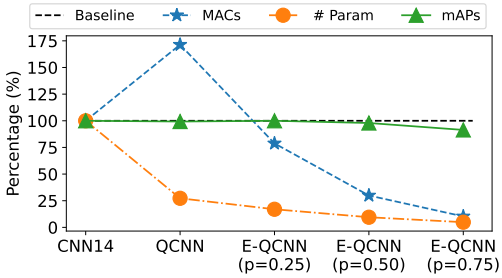


Figure 3: *Relative comparative performance of QCNN and E-QCNN for various pruning rates with baseline CNN14 model. Here, 100% is equivalent to baseline performance with MACs (21G), #Param (81M), mAPs (0.431).*

loss, ADAM optimizer, learning rate (LR) of 1e-3, batch size set to 32 using a single NVIDIA RTX 3090-24GB GPU.

## 5. Results and Analysis

### 5.1. Comparison with the baseline system

We first compare the baseline CNN14 with QCNN and E-QCNN in terms of mAPs, model parameters and MACs. The results of this experiment are reported in Figure 3. It can be observed that transforming CNN14 into the quaternion domain (QCNN) reduces parameter count by approximately 73% while achieving competitive performance with only 0.3 percentage points reduction in mAPs. However, this comes at approximately a 1.6× increase in computational complexity for the QCNN model compared to that of CNN14. As expected, the computational complexity is addressed in E-QCNN, where a good trade-off is achieved between the pruning rate and mAPs. For instance, pruning 25% quaternion filters from QCNN reduces parameter count and MACs by 83% and 25%, respectively, without any performance loss. Similarly, pruning 50% quaternion filters reduce parameter count and MACs by 90% and 70%, respectively, with only a 0.9 percentage points reduction in mAPs compared to that of CNN14. On the other hand, pruning 75% quaternion filters from QCNN reduces the performance by 3.6 percentage points at the expense of reduction in the parameter count and the MACs by 95% and 90%, respectively, compared to that of CNN14.

In addition, we analysed the convergence behaviour of the proposed compressed QCNN models. As demonstrated in Figure 4, QCNN/E-QCNN models exhibit stable convergence over epochs during training or fine-tuning stages. Such a behaviour is often challenging to achieve for structurally compressed models in practice [17], which motivates the use of quaternion models in other speech/audio applications in future.

### 5.2. Comparison with existing models

In this experiment, we compare the performance trade-off (mAPS vs parameter count) of the proposed QCNN models and existing popular, efficient architectures. The results of this experiment are reported in Figure 5. It can be observed that among the top-performing ResNet & CNN14 models, E-QCNN ($p = 25\%$) has the lowest parameter count. Similarly, among existing low-footprint models, namely MobileNetV1/V2, CNN6/10, EfficientNet-B2 and DeepRes models, E-QCNN ($p = 75\%$) has the best mAPs and lowest parameter count. MobileNetV3, an advanced architecture optimized using computationally expensive neural architecture search, exhibits a slightly higher performance (mAPs) at the expense of a few more parameters. It is worth highlighting that for a similar parameter budget, the proposed E-QCNN achieves better performance (mAPs) than E-PANNs (pruned only CNN14) and low-footprint CNN6 models. This suggests the benefits of obtaining a low-footprint and efficient CNN from a large-size CNN by applying both quaternion transformation and pruning compared to applying pruning alone or training similar-size CNN from scratch.

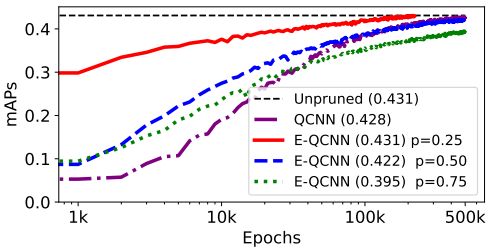


Figure 4: *mAPs obtained while training or fine-tuning process for QCNN and pruned QCNN at different p. mAPs inside ()*

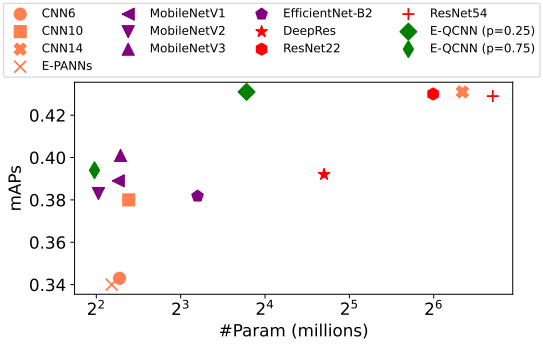


Figure 5: *mAPs versus parameter count for various models.*

## 6. Conclusion

In this study, we introduce Quaternion neural networks optimized for efficient audio tagging on devices constrained by computational and memory capacities. We illustrate how quaternion models adeptly capture complex interactions across various perspectives of the input data, outperforming their multichannel counterparts typically used in traditional models. Through experiments conducted on the AudioSet dataset, we demonstrate that quaternion models attain similar levels of performance with markedly fewer parameters compared to other models achieving equivalent accuracy. Furthermore, we explore the application of pruning techniques within Quaternion Convolutional Neural Networks (QCNNs) to further reduce the model size without significantly impacting its ability to accurately tag audio content, thereby enhancing the feasibility of deploying advanced audio analysis models on resource-limited devices.

## 7. Acknowledgements

This work was partly supported by JPMC faculty award, the Infosys Foundation via Infosys Centre for AI, IIIT Delhi, India and Engineering and Physical Sciences Research Council (EPSRC) Grant EP/T019751/1 "AI for Sound (AI4S)". This publication is supported by openly available public dataset at locations referenced in this paper. For the purpose of open access, the authors have applied a Creative Commons Attribution

(CC BY) licence to any Author Accepted Manuscript version arising.

## 8. References


[1] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018.

[2] V. Abrol and P. Sharma, "Learning hierarchy aware embedding from raw audio for acoustic scene classification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1964–1973, 2020.

[3] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "AudioSet: An ontology and human-labeled dataset for audio events," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 776–780, 2017.

[4] W. Su, L. Li, F. Liu, M. He, and X. Liang, "AI on the Edge: a comprehensive review," *Artificial Intelligence Review*, vol. 55, no. 8, pp. 6125–6183, 2022.

[5] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018.

[6] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[7] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[8] F. Lagunas, E. Charlaix, V. Sanh, and A. Rush, "Block pruning for faster transformers," in *Conference on Empirical Methods in Natural Language Processing*, November 2021, pp. 10 619–10 629.

[9] M. Xia, Z. Zhong, and D. Chen, "Structured pruning learns compact and accurate models," in *Association for Computational Linguistics (ACL)*, 2022, pp. 1513–1528.

[10] X. Zhu, Y. Xu, H. Xu, and C. Chen, "Quaternion convolutional neural networks," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 631–647.

[11] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," *International Conference on Learning Representations (ICLR)*, 2017.

[12] S. Kumar, R. K. Singh, and A. Chaudhary, "A novel non-linear neuron model based on multiplicative aggregation in quaternionic domain," *Complex & Intelligent Systems*, vol. 9, no. 3, pp. 3161–3183, June 2023.

[13] X. Qiu, T. Parcollet, M. Ravanelli, N. D. Lane, and M. Morchid, "Quaternion neural networks for multi-channel distant speech recognition," *Interspeech*, 2020.

[14] T. Parcollet, M. Morchid, and G. Linarès, "Quaternion convolutional neural networks for heterogeneous image processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 8514–8518.

[15] T. Parcollet, Y. Zhang, M. Morchid, C. Trabelsi, G. Linares, R. de Mori, and Y. Bengio, "Quaternion Convolutional Neural Networks for End-to-End Automatic Speech Recognition," in *Interspeech*, 2018, pp. 22–26.

[16] D. Comminiello, M. Lella, S. Scardapane, and A. Uncini, "Quaternion convolutional neural networks for detection and localization of 3d sound events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2019, pp. 8533–8537.

[17] A. Chaudhary and V. Abrol, "Towards on-device keyword spotting using low-footprint quaternion neural models," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 1–5, 2023.

[18] Y. Gong, A. Rouditchenko, A. H. Liu, D. Harwath, L. Karlinsky, H. Kuehne, and J. Glass, "Contrastive audio-visual masked autoencoder," in *International Conference on Learning Representations (ICLR)*, 2022.

[19] Y. Gong, Y.-A. Chung, and J. Glass, "PSLA: improving audio tagging with pretraining, sampling, labeling, and aggregation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3292–3306, 2021.

[20] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. De Freitas, "Predicting parameters in deep learning," *Advances in Neural Information Processing Systems*, pp. 2148–2156, 2013.

[21] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *International Conference on Learning Representations (ICLR)*, 2019.

[22] J.-H. Luo, H. Zhang, H.-Y. Zhou, C.-W. Xie, J. Wu, and W. Lin, "ThiNet: pruning CNN filters for a thinner net," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2525–2538, 2018.

[23] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2074–2082, 2016.

[24] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 243–254, 2016.

[25] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann, "Towards optimal structured CNN pruning via generative adversarial learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2758–2795.

[26] J.-H. Luo and J. Wu, "AutoPruner: An end-to-end trainable filter pruning method for efficient deep model inference," *Pattern Recognition*, vol. 107, p. 107461, 2020.

[27] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao, "HRank: Filter pruning using high-rank feature map," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1529–1538, 2020.

[28] S.-K. Yeom, K.-H. Shim, and J.-H. Hwang, "Toward compact deep neural networks via energy-aware pruning," in *tinyML Research Symposium*, 2021.

[29] A. Singh, H. Liu, and M. D. Plumbley, "E-PANNs: sound recognition using efficient pre-trained audio neural networks," in *Internoise*, Chiba, Greater Tokyo, Japan, 2023.

[30] T. Parcollet, M. Morchid, and G. Linarès, "Deep quaternion neural networks for spoken language understanding," *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 504–511, December 2017.

[31] S. Kumar, A. Chaudhary, and R. K. Singh, "On the learning machine with amplificatory neuron in complex domain," *Arabian Journal for Science and Engineering*, vol. 45, no. 12, pp. 10 287–10 309, Dec 2020.

[32] C. J. Gaudet and A. S. Maida, "Deep quaternion networks," in *International Joint Conference on Neural Networks (IJCNN)*, 2018.

[33] L. Ford, H. Tang, F. Grondin, and J. R. Glass, "A deep residual network for large-scale acoustic scene analysis." in *Interspeech*, 2019, pp. 2568–2572.

[34] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for MobileNetV3," *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1314–1324, 2019.

[35] F. Schmid, K. Koutini, and G. Widmer, "Efficient large-scale audio tagging via transformer-to-CNN knowledge distillation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[36] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning (ICML)*. PMLR, 2019, pp. 6105–6114.