Embedded Face Recognition for Personalized Services in the Assistive Robotics

Iris Walter[1], Jonas Ney[2], Tim Hotfilter[1], Vladimir Rybalkin[2], Julian Hoefer[1], Norbert Wehn[2], and Jürgen Becker[1]

[1] Karlsruhe Institute of Technology {iris.walter, hotfilter, julian.hoefer, becker}@kit.edu

[2] Technical University of Kaiserslautern {ney, rybalkin, wehn}@eit.uni-kl.de

Abstract. Recently, the field of assistive robotics has drawn much attention in the health care sector. In combination with modern machine learning-supported person recognition systems, they can deliver highly personalized services. However, common algorithms for person recognition such as convolutional neural networks (CNNs) consume high amounts of power and show low energy efficiency when executed on general-purpose computing platforms.
In this paper, we present our hardware architecture and field programmable gate array (FPGA) accelerator to enable on-device person recognition in the context of assistive robotics. Therefore, we optimize a neural network based on the SqueezeNet topology and implement it on an FPGA for a high degree of flexibility and reconfigurability. By pruning redundant filters and quantization of weights and activations, we are able to find a well-fitting neural network that achieves a high identification accuracy of 84 %. On a *Xilinx Zynq Ultra96v2*, we achieve a power consumption of 4.8 W, a latency of 31 ms and an efficiency of 6.738 FPS/W. Our results outperform the latency by 1.6x compared to recent person recognition systems in assistive robots and energy efficiency by 1.7x for embedded face recognition, respectively.

Keywords: Ambient assisted living · Assistive robotics · Convolutional neural networks · Face recognition · Field programmable gate array · Quantization

1 Introduction

Assistive robotics promise major enhancements in the health and elderly care sector. In an ambient assisted living (AAL) where several people live at the same place, assistive services include, for example, reminder and help with medication, support in rehabilitation, calling for help in emergency cases or needs-based help with daily living. For this purpose, assistive robots must be able to understand their environment, assess interaction possibilities with persons in their vicinity and adapt to their personal needs. While movements and environment perception are still big challenges to overcome, the tremendous improvements of deep neural networks (DNNs) in recent years might help to address them. The emergence

of convolutional neural networks (CNNs), which take spatial information into account and hence show very good performance in image processing, vision and perception, confirms this trend further. Their performance and prediction results even surpass human vision as well as traditional computer vision algorithms, for instance in cancer detection [3]. Although Machine Learning (ML) algorithms are considered enablers for those challenging use cases, they come with a very high computational complexity, typically resulting in high power consumption when implemented on general purpose processors. In addition, those throughput oriented architectures fail to meet the low latency requirements. Thus, many currently available off the shelf solutions such as central processing units (CPUs) or graphics processing units (GPUs) cannot satisfy the requirements for embedded applications.

For this reason, a shift towards dedicated DNN hardware accelerators can be observed. With dedicated accelerators, co-design in accordance with the algorithm is conceivable [9]. For instance, optimization strategies like quantization [11] or pruning [8] can improve neural network inference performance 100-fold and energy efficiency 1000-fold compared to a CPU baseline implementation. Quantization aims at a reduction of the bit precision of data and weights, thus it reduces the computational complexity. Pruned neural network topologies can reduce the total amount of operations through smart bypass techniques of superfluous operations during execution or by removing redundant connections in the network beforehand, respectively.

While hardware accelerators for DNNs can achieve vast improvements in the inference, dedicated hardware structures usually lack flexibility. The trade-off between accuracy and energy efficiency cannot be fully exploited when the number format and bitwidth are fixed. Accelerators on Field programmable gate arrays (FPGAs) [2] on the other hand, are highly customizable and offer great potential for optimization to balance numerical precision and desired accuracy for a specific use case.

In this paper, we present an FPGA-based accelerator for face recognition focusing on privacy and low power consumption to enable personalized services in assistive robotics. In summary, our contributions are threefold:

- For the application in the field of personalized assistive robotics, we choose and derive a hardware-aware CNN topology. This model emphasizes low power for mobile deployment and low latency for high user acceptance.
- We implement the network with an HLS-based tool-flow on a *Xilinx* FPGA and compare different configurations in terms of accuracy, resource usage and energy efficiency.
- We select and evaluate the optimal configuration regarding the trade-off between resource utilization and accuracy. Based on this model, we demonstrate low power person recognition with a latency of 31 ms and an energy efficiency of 6.7 FPS/W, outperforming recent work by 1.6x regarding latency [5] and 1.7x regarding energy efficiency [32].

2 Related Work

2.1 Personalization in Assistive Robotics

AAL methods, targeting elderly and other people who are dependent on care, support and help people in daily life. Exemplary, they can improve the safety of a person or assist rehabilitation. Personalization of the provided service as well as continuous adaption to the constantly changing needs of the patient are identified as key factors for user acceptance and satisfaction with an assistive robot [20].

Ghiţă et al. [7] present a people detection and recognition of a *Pepper* robot to show reminders to people in an AAL. Locally captured images of the environment of the robot are transmitted to a remote machine where people are recognized using FaceNet [22]. Their person recognition takes 300 ms.

Duque et al. [5] implemented a gaze control for a robot head to improve interaction with persons. To determine the person to look at, they combine face recognition, pose recognition and speech detection. For face recognition, they use a ResNet with 29 convolutional layers inferencing in less than 0.05 s on a *GPU GeForce RTX 2080 Ti*.

2.2 Face Recognition

Face recognition tasks have been widely studied for several decades. In general, the common algorithms for face recognition can be divided into three different categories: holistic methods, local methods and deep learning (DL) methods. Most holistic methods are based on Eigenfaces, i.e., a set of eigenvectors representing images in a lower dimensional space, and were first proposed by Sirovich and Kirby [23]. Local or feature-based methods for face recognition aim to find similar local features from the face or from special regions of the face. As a result, they are more robust to variations like pose, viewpoint and expression, compared to holistic methods [31]. DL methods are based on DNNs achieving state-of-the-art results in various image processing tasks, and have also drawn attention in the field of face recognition in recent years. By making use of a large set of training samples, DNNs are able to learn how and which features to extract from an input image to increase recognition accuracy significantly compared to holistic and local methods. The first breakthroughs of DL-based algorithms for face recognition were achieved by DeepFace [25] and Deep ID [24] in 2014. Since then, the state-of-the-art performance has been improved drastically, mainly driven by the advancements made in DL, for instance, Google's FaceNet accomplished an accuracy of over 99 % on the labeled faces in the wild (LFW) benchmark. Overall, the accuracy has been increased from around 90 % for the classical methods, up to around 99.8 % [21] [18] in just three years [27].

2.3 Low-complexity Neural Networks

In recent years, one main focus of DNN research was increasing accuracy without considering the size and complexity of the network. This becomes a problem, es-

pecially in embedded devices where both latency and power requirements need to be fulfilled. Thus, there is an increasing demand for neural network architectures that are suitable for low power embedded devices.

One of the first presented DNNs optimized for the embedded domain is SqueezeNet, introduced in 2016 [13]. It achieves an accuracy on the same level as AlexNet [15] for the ImageNet [4] dataset with 50 times less trainable parameters. SqueezeNet consists of 18 convolutional layers with 1.25 M parameters. The main building block of the network is the fire module that consists of squeeze layers with 1x1 convolutional kernels and expand layers with a mix of 1x1 and 3x3 kernels.

In 2017, Howard et al. presented MobileNets which are CNNs for mobile vision application, especially focusing on low latency and low power applications [10]. The architecture is based on depthwise separable convolutions to reduce computational complexity while increasing the model depth. MobileNets accomplish a top-1 accuracy of 70.6 % for ImageNet with only 4.2 M trainable parameters.

Shortly later, in 2018, ShuffleNet was proposed [30], which is an extremely efficient neural network designed for mobile devices. It makes use of pointwise group convolution and channel shuffle operations to reduce computation cost while maintaining accuracy. A ShuffleNet unit is used to enable information flow across feature channels by shuffling input data of different groups. As a result, ShuffleNet obtains a three percentage points higher top-1 accuracy than MobileNet on ImageNet while, at the same time, achieving a 13x speedup compared to AlexNet. However, ShuffleNet is composed of 50 layers with 5.4 M parameters and therefore requiring 4.3x more memory space than SqueezeNet.

2.4 FPGA Implementations of Face Recognition

Various FPGA implementations of DNN-based face recognition systems have been proposed recently.

Zhunge et al. implemented a DNN for face recognition based on FaceNet using high-level synthesis (HLS) on a *Xilinx Virtex Ultrascale+* device [32]. They explored multiple fast convolution algorithms to find an optimal strategy to apply them to the different types of convolutions of the model. Their FPGA accelerator accomplishes a single face recognition in 23.7 ms with a power consumption of 10.6 W. As a result, they outperform the *Nvidia GTX 1080 GPU* with respect to latency by a factor of 3.75 and achieve an energy efficiency of 3.981 FPS/W.

In 2021, Liu et al. proposed a heterogeneous computing system with a hardware accelerator based on MobileNet that is used as face tracking platform [17]. The FPGA accelerator is integrated into a CPU and GPU based system that is aided by a delay-aware, energy-efficient scheduling algorithm. Their design, implemented on the *Intel Stratix 10*, achieves a latency of only 1.3 ms per image. They claim to provide a low power solution, although power consumption never is reported but only the power efficiency of their system. Derived from this value

and the complexity of MobileNet, the power consumption can be estimated to be around 26 W.

In summary, state-of-the-art face recognition implementations based on DNNs achieve latencies that are sufficient for real-time applications. However, none of the works provide solutions that are suitable for battery-powered embedded devices with respect to power consumption, while still fulfilling real-time constraints. They have either very high power consumption [17] or provide only moderate latency [32]. Thus, we focus on a DNN-based face recognition implementation that is real-time suitable as well as consuming low power for the CASIA-Webface dataset [28].

2.5 FINN

To reduce development time and to guarantee compatibility to other designs, the hardware implementation presented in this work is based on the open-source hardware library [16] which is part of Xilinx FINN framework [26]. FINN is a tool to explore the design space of DNN inference accelerators on FPGAs. The DNN layers of the FINN hardware library are designed as dataflow architecture, resulting in high-throughput and low latency implementations. Furthermore, the framework provides an adjustable degree of parallelization for each layer, which can be used to find an optimal trade-off between resource usage and latency. Additionally, FINN uses fixed-point format that allows for a higher level of flexibility and enables customizable quantization.

3 Context and Methodology

In a typical AAL, several people live at the same place, hence an assistive robot needs a real-time scene perception to locate and interact with persons in a room. To perform personalized assistive services, the robot must first and foremost identify the people in its workspace. Recent work already proposed face recognition techniques for assistive robots, but lack either privacy due to remote computation [7] or adaption to low power requirements of embedded systems [5].

We use the assistive robot *ARMAR-6* [1] which is equipped with one depth camera and two stereo cameras for peripheral and foveal vision, respectively. The camera images are forwarded to a face detection unit, e.g. realized with an MTCNN [29] that is a commonly used CNN to detect bounding boxes of objects or faces. A typical face detection with an MTCNN on a GPU takes 122 ms [6]. Afterwards the detected faces are forwarded to a face recognition unit.

In this paper, we focus on the real-time face recognition and its efficient implementation on the robot with low power consumption to address limited power supply in the embedded, battery powered application. We implement the accelerator entirely on an FPGA board to enable on-device execution of face recognition and thereby enhance privacy. This platform furthermore facilitates updates and allows reconfiguration during runtime to supply other algorithms that provide the assistive services from the FPGA.

3.1 Model Topology

As DL methods achieve very good accuracy results on the task of face recognition, our approach is based on a DNN. Selecting the DNN topology is a crucial design decision for embedded face recognition, as it determines the initial architecture and performance optimization potential of the later hardware implementation. SqueezeNet [13], MobileNet [10] and ShuffleNet [30] are good candidates for embedded face recognition, since they provide a relatively small computational complexity as well as sufficient accuracy on the ImageNet dataset. As SqueezeNet has the smallest memory footprint with 1.25 M parameters and has no residual paths or skip connections that increase implementation complexity, this topology suites excellent for the hardware implementation of face recognition in assistive robotics. In addition, it has a high reduction potential which was shown in the original paper [13].

Starting from the original SqueezeNet v1.1 topology [12], we apply different optimizations: First, we reduce the number of filters of the last convolution to 256 instead of 1,000. This decreases the output dimensionality and still enables face recognition in environments with a relatively small group of persons for instance in the aforementioned application. For regularization and to avoid overfitting, we insert a dropout layer before the last convolution. Additionally, we replace the final softmax layer with a convolutional layer with 128 filters and train with triplet loss [22] to learn a multi-dimensional embedding for faces. We implement this variant of SqueezeNet in PyTorch. To reduce the model size and computational complexity, we apply an unconditional filter pruning by factor two that is proved to be as effective as structured filter pruning [19], i.e., half the number of filters compared to the original SqueezeNet v1.1 is applied per convolutional layer. This pruning quarters the model size with a negligible accuracy degradation of less than one percentage point for face recognition on the CASIA-Webface dataset [28].

Moreover, we insert batch normalization layers after each expand convolutional layer of the fire module to stabilize training, see Figure 1. During inference,

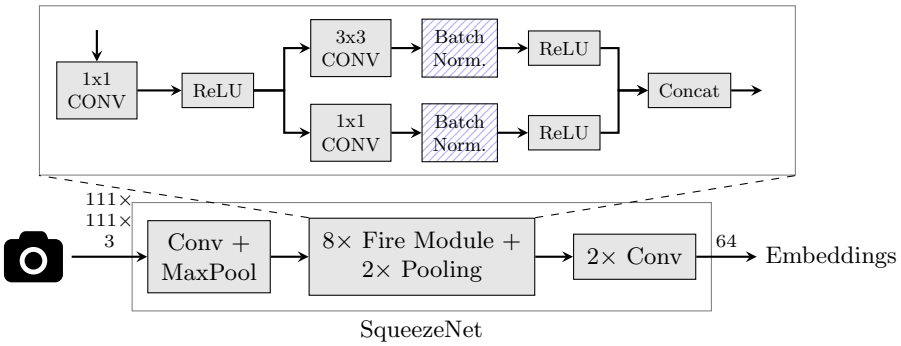


Fig. 1: Overview of our SqueezeNet, changes are highlighted by a dashed pattern.

we fold the batch normalization parameters into the previous convolutional layer as proposed in [14] to reduce computational complexity further.

3.2 Hardware Architecture

We use the hardware library of FINN [16] to implement an HLS model of SqueezeNet. For synthesis and deployment of the HLS model on the FPGA board, we use *Xilinx Vivado (HLS)*. To avoid offloading intermediate feature maps between the layers, we implement the entire CNN as dataflow architecture on the chip. A dataflow architecture can be considered as task-level pipelining, i.e., the results of one layer are forwarded to the next before the computation of the entire layer has finished. Therefore, the subsequent layer starts processing as soon as the data for the first calculation step is available and the need for buffering intermediate results between the layers decreases.

The total inference latency of the CNN is determined by the first convolutional layer which we observed being the bottleneck layer and thus has to be maximally unrolled. We unroll the other layers by the minimum factor that does not increase this latency. Additionally, we reduce computational complexity, hence the resource utilization, by quantization of the weights and activations of the SqueezeNet with fixed point datatypes instead of floating point. To determine an appropriated quantization scheme which covers the whole range of values, we profile the weight and activation floating point values of our trained model and choose minimal integer bit widths for each layer. As the fraction bit width influences accuracy, it is part of the design space exploration (DSE). We further add a user-level driver for communication between the hardware and software on the board.

4 Evaluation and Discussion

4.1 Experimental Setup and Training

Initially, we train our CNN with 32-bit floating point on the CASIA-Webface dataset [28]. $2,115$ of $10,575$ identities are assigned to the test dataset and the images of the remaining $8,460$ identities are split by 80 % into training data and by 20 % into validation data. The learning rate is set to 0.05, dropout to 0.5, momentum to 0.9, batch size to 400 and our pruned SqueezeNet is trained for 500 epochs. To evaluate the identification accuracy in the presented use case, we create a small test dataset from the unseen test data including ten identities with ten images each. We calculate a reference embedding for each identity corresponding to the mean of the embedding of five different images. For identification of a person, an image passes the SqueezeNet and the euclidean distance to the reference embeddings of each identity is determined. The identity related to the closest reference embedding is recognized by the network. On our test dataset, the software model achieves 89.6 % accuracy with full precision.

After batch normalization folding, our CNN has $222,940$ parameters and 17.47 M MAC operations are executed to process one image. Compared to the

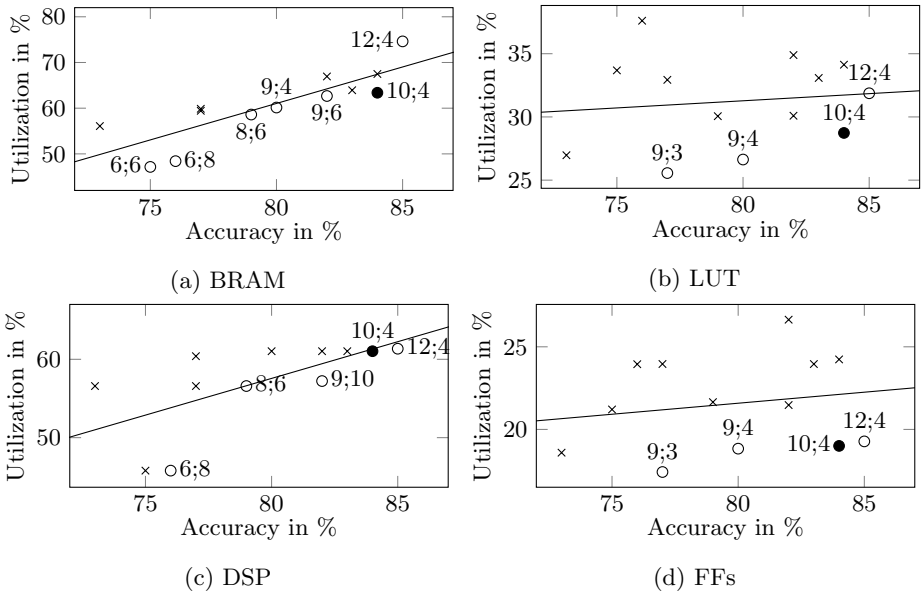


Fig. 2: Utilization of BRAMs, LUTs, DSPs and Flip-Flops of the *ZCU102* in relation to the accuracy. Pareto optimal configurations are shown as circles and are labeled with the quantization step: The first and latter number represents the fraction bit width of weights and activations, respectively. A trend line indicates the correlation between primitive utilization and accuracy.

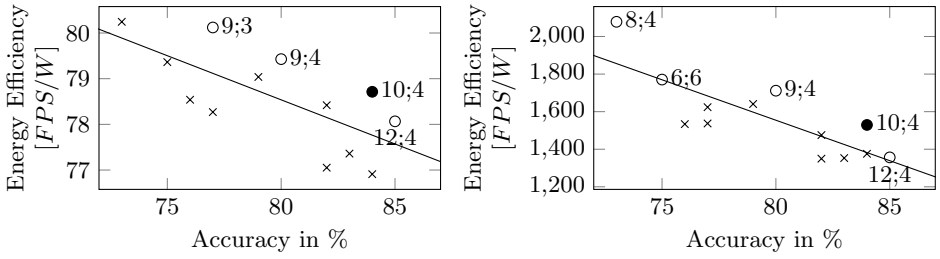
original SqueezeNet v1.1, the computational complexity of our variant decreases by factor 20 and the model size by factor 5.

The *Xilinx ZCU102* with large FPGA resources allows for a wide range of variation in the optimization parameters of the accelerator. This is why we choose the *ZCU102* as target platform for the DSE. For later deployment of a low power accelerator for face recognition, we choose the well-fitting *Xilinx Ultra96v2* which provides four times less resources and lower static power consumption.

4.2 Design Space Exploration

For embedded face recognition, we analyze the influence of different quantization schemes on the trade-off between accuracy and resource utilization which directly affects energy efficiency. Therefore, we apply individual quantization schemes for each layer, i.e., we use minimal integer bit widths as described in subsection 3.2 and apply different fixed fraction bit widths for weights and activations. For example, we denote 10;4 as the quantization scheme with ten fraction bits for all weights and four fraction bits for all activation values.

The design is unrolled to the degree which accomplishes a minimal latency of $27,225$ *cycles*. For this configuration, we report the accuracy, resource utilization and energy efficiency for different quantization schemes on the *ZCU102*,



(a) Total power (b) Dynamic power

Fig. 3: Total and dynamic energy efficiency on the *ZCU102* in relation to the accuracy. Pareto optimal configurations are shown as circles and are labeled with the quantization level. A trend line indicates the correlation between the efficiency and the accuracy.

see Figure 2 and Figure 3. For less precision in the number representation of weights and activations, less resources are required, energy efficiency increases and the identification accuracy decreases. As it is shown in Figure 2a, the BRAM utilization dominates the total resource utilization and is effected most by the varying quantization scheme. Hence, BRAMs are the limiting resource which determine feasibility of the accelerator on a certain FPGA.

The configuration 10;4 with ten weight fraction bits and four activation fraction bits is a Pareto-optimal design with respect to all hardware costs, which achieves a moderate accuracy drop of five percentage points compared to the software model with full precision and without batch normalization folding. This model requires 63.4 % of the BRAM resources of the *ZCU102* and is selected for further considerations.

For power evaluation, we first measure static power of the FPGA board with a digital multimeter and then the total power consumed to process face recognition. Dynamic power consumption is reported as the difference between these measurements. At a clock frequency of 300 MHz, the accelerator executes a single face recognition in 0.481 ms and draws 26.423 W in total. As a result, the accelerator achieves a total energy efficiency of 78.713 FPS/W, see Table 1. Compared to [17], our unrolled accelerator on a *ZCU102* is 2.7 times faster and more energy efficient, respectively. If the HLS model is implemented as a full sequential architecture, the model requires 19.1 % of the BRAM resources of a *ZCU102* and realizes a single face recognition in 25.338 ms at the same clock frequency. Therefore, the total energy efficiency of the rolled model is 1.667 FPS/W on a *ZCU102*.

For very high throughput, the unrolled implementation might be preferred. However, on the *ZCU102*, we could show that the real-time constraints of a robot scene perception can also be satisfied with the rolled model. As this model has a smaller resource footprint, we intend the implementation on a smaller FPGA to reduce power and energy consumption further. An *Ultra96v2* has four times

Table 1: Power consumption and energy efficiency of different configurations of our SqueezeNet accelerator and other embedded face recognition.

Platform	Runtime [ms]	Power consumption		Throughput [FPS]	Energy efficiency	
		Total [W]	Dynamic [W]		Total [FPS/W]	Dynamic [FPS/W]
Stratix 10 [17]	1.310	26	-	763.5	29.586	-
VU9P [32]	23.7	10.6	-	42.194	3.981	-
ZCU102[1]	**0.481**	26.423	1.360	**2,079.824**	**78.713**	**1,529.282**
ZCU102[2]	25.338	23.679	0.279	39.467	1.667	141.457
Ultra96v2[3]	30.712	**4.832**	**0.236**	32.560	6.738	137.967

[1]unrolled at 300 MHz, [2]rolled at 300 MHz, [3]rolled at 250 MHz

less BRAMs than a *ZCU102* which suites well for the implementation of the rolled model. We synthesize the rolled model for the *Ultra96v2* at a maximum clock frequency of 250 MHz resulting in a BRAM resource utilization of 80.8 %. On the *Ultra96v2*, our implementation inferences in 30.71 ms which is 1.6 times faster than the face recognition of [5]. Moreover, we measure a dynamic power consumption of 0.236 W and a total power consumption of 4.832 W which is five times lower than on the *ZCU102* and less than half the power consumption of [32]. The total energy efficiency on the *Ultra96v2* is four times better than on the *ZCU102* and 1.7 times better than in [32], respectively, see Table 1.

Considering 122 ms of processing time for the face detection unit [6], the entire person identification is realized within 152 ms using the *Ultra96v2* for low power face recognition. This is in the range of human reaction time and thus natural interaction with the *ARMAR-6* is enabled.

5 Conclusion and Future Work

In this paper, we present an embedded low power face recognition, which enables real-time person recognition for assistive robotics. Due to on-device execution, personalized services can be provided at the edge and privacy is assured. Therefore, we optimize a CNN topology for the design of an FPGA accelerator and analyze the performance-accuracy trade-off on a *Xilinx* FPGA depending on the quantization level. Our implementation on an *Ultra96v2* accomplishes a single face recognition in 31 ms consuming only 4.8 W. We achieve 6.738 FPS/W energy efficiency which outperforms the state-of-the-art 1.7-fold. In the future, we plan to implement the whole processing chain of scene perception, including face detection and the execution of personalized assistive services.

References

1. Asfour, T., Waechter, M., Kaul, L., Rader, S., Weiner, P., Ottenhaus, S., Grimm, R., Zhou, Y., Grotz, M., Paus, F.: Armar-6: A high-performance humanoid for human-robot collaboration in real-world scenarios. IEEE Robotics & Automation Magazine **26** (2019)
2. Baehr, S., McCarney, S., Meggendorfer, F., Poehler, J., Skambraks, S., Unger, K., Becker, J., Kiesling, C.: Low latency neural networks using heterogenous resources on fpga for the belle ii trigger. CoRR **abs/1910.13679** (2019)
3. Brinker, T.J., Hekler, A., Enk, A.H., Klode, J., Hauschild, A., Berking, C., Schilling, B., Haferkamp, S., Schadendorf, D., Holland-Letz, T., et al.: Deep learning outperformed 136 of 157 dermatologists in a head-to-head dermoscopic melanoma image classification task. European Journal of Cancer **113**, 47–54 (2019)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
5. Duque-Domingo, J., Gómez-García-Bermejo, J., Zalama, E.: Gaze control of a robotic head for realistic interaction with humans. Frontiers in Neurorobotics (2020)
6. Esler, T.: Face recognition using pytorch. https://github.com/timesler/facenet-pytorch, (Accessed on 07/06/2021)
7. Ghiţă, Ş.A., Barbu, M.Ş., Gavril, A., Trăscău, M., Sorici, A., Florea, A.M.: User detection, tracking and recognition in robot assistive care scenarios. In: Annual Conference Towards Autonomous Robotic Systems. Springer (2018)
8. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both weights and connections for efficient neural networks. CoRR **abs/1506.02626** (2015)
9. Hotfilter, T., Kempf, F., Becker, J., Reinhardt, D., Baili, I.: Embedded image processing the european way: A new platform for the future automotive market. In: World Forum on Internet of Things. p. 1–6 (2020)
10. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR **abs/1704.04861** (2017)
11. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Quantized neural networks: Training neural networks with low precision weights and activations. CoRR **abs/1609.07061** (2016)
12. Iandola, F.N.: Github forresti/squeezenet/squeezenet_v1.1/. https://github.com/forresti/SqueezeNet/tree/master/SqueezeNet_v1.1 (2016), (Accessed on 02/01/2021)
13. Iandola, F.N., Moskewicz, M.W., Ashraf, K., Han, S., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and $<$ 1mb model size. CoRR **abs/1602.07360** (2016)
14. 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: IEEE Conference on Computer Vision and Pattern Recognition (2018)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems. vol. 25. Curran Associates, Inc. (2012)

16. Labs, X.R.: Finn hls library. https://github.com/Xilinx/finn-hlslib, (Accessed on 02/22/2021)
17. Liu, X., Yang, J., Zou, C., Chen, Q., Yan, X., Chen, Y., Cai, C.: Collaborative edge computing with fpga-based cnn accelerators for energy-efficient and time-aware face tracking system. IEEE Transactions on Computational Social Systems (2021)
18. Liu, Y., Li, H., Wang, X.: Rethinking feature discrimination and polymerization for large-scale recognition. CoRR **abs/1710.00870** (2017)
19. Liu, Z., Sun, M., Zhou, T., Huang, G., Darrell, T.: Rethinking the value of network pruning. In: International Conference on Learning Representations (2018)
20. Matarić, M.J., Scassellati, B.: Socially assistive robotics. Springer handbook of robotics pp. 1973–1994 (2016)
21. Ranjan, R., Castillo, C.D., Chellappa, R.: L2-constrained softmax loss for discriminative face verification. CoRR **abs/1703.09507** (2017)
22. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: IEEE Conference on Computer Vision and Pattern Recognition (2015)
23. Sirovich, L., Kirby, M.: Low-dimensional procedure for the characterization of human faces. J. Opt. Soc. Am. A **4**(3), 519–524 (1987)
24. Sun, Y., Wang, X., Tang, X.: Deep learning face representation from predicting 10,000 classes. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
25. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1701–1708 (2014)
26. Umuroglu, Y., Fraser, N.J., Gambardella, G., Blott, M., Leong, P., Jahre, M., Vissers, K.: Finn: A framework for fast, scalable binarized neural network inference. In: ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. pp. 65–74. ACM (2017)
27. Wang, M., Deng, W.: Deep face recognition: A survey. CoRR **abs/1804.06655** (2018)
28. Yi, D., Lei, Z., Liao, S., Li, S.Z.: Learning face representation from scratch. CoRR **abs/1411.7923** (2014)
29. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters **23**(10) (2016)
30. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. CoRR **abs/1707.01083** (2017)
31. Zhou, H., Mian, A., Wei, L., Creighton, D., Hossny, M., Nahavandi, S.: Recent advances on singlemodal and multimodal face recognition: A survey. IEEE Transactions on Human-Machine Systems **44**(6), 701–716 (2014)
32. Zhuge, C., Liu, X., Zhang, X., Gummadi, S., Xiong, J., Chen, D.: Face recognition with hybrid efficient convolution algorithms on fpgas. CoRR **abs/1803.09004** (2018)