Fast Emotion Recognition Neural Network for IoT Devices

S. Mikhaylevskiy, V. Chernyavskiy, V. Pavlishen, I. Romanova,
HSE University
Moscow, Russia
{smikhaylevskiy, vchernyavskiy, vspavlishen}@edu.hse.ru,
iromanova@hse.ru

R. Solovyev
Institute for Design Problems in Microelectronics of
Russian Academy of Sciences (IPPM RAS)
Moscow, Russia
turbo@ippm.ru

***Abstract*—The last decades have witnessed rapid IoT technologies development, which provided ubiquitous human-computer interactions. Building intelligent systems of various types, among which emotion recognition systems, is important challenge nowadays. Especially pressing problem is to build a real-time portable system which can be embedded in low performance hardware. We propose a high accuracy emotion recognition system, which can be deployed on a single board Raspberry Pi computer to perform real-time recognition of 4 facial expressions: neutral, angry, surprised, and happy. Recognition pipeline is divided into two main stages: human face detection and facial expression classification. Both stages are performed by deep neural networks with simple yet effective design. Several optimization techniques, such as weights quantization and model tracing, were applied after model training, to gain extra execution time reduction. The introduced system is lightweight and fast, is executed locally on a cost-effective single board computer, and requires minimum resources to make and transmit predictions, what makes proposed system an effective IoT device.**

Keywords—computer vision, artificial neural networks, emotion recognition, neural networks optimization, Raspberry Pi, Internet of Things (IoT)

I. Introduction

In modern world, mechanization is being replaced by a new dominant trend – process automation. With the development of artificial intelligence, especially neural network technologies, it became possible to automate processes that are very hard to automate algorithmically, such as sentiment analysis or speech recognition. At the same time, the ability to communicate effectively with people is an integral part of the success of many types of professional or everyday activities. Therefore, solving the problem of automating the process of recognizing emotions becomes important for many areas in which direct interaction with people is necessary. Also, in some technological areas, such as robotics or Internet of Things, it is required to carry out recognition under conditions of limited computing resources and in real time, therefore it turns out to be necessary to create a portable, high-speed emotion recognition system.

We propose a composed deep learning model for human face detection and emotion classification, which can be deployed on a single board Raspberry Pi computer to perform real-time emotion recognition.

II. Related works

Human facial expressions are a universal and powerful way for the demonstration of his emotional state that significantly affects his current model of behavior. Many works and studies attempted to solve the problem of automating the emotion recognition process, because of its practical importance for many areas, such as robotics, medicine and psychology, management.

After Ekman and Friesen showed [1] that there is a set of basic cultural-, gender- and age-independent human facial expressions, rapid development of emotion recognition methods began. For a long time, pattern recognition methods of classical computer vision were used for visual emotion recognition. Specifically, to compose vector representations of facial expressions data dimensionality reduction techniques were used, such as the principal component method, discrete cosine transform, Haar transform [2], [3], [4], [5]. This approach suffered from poor generalization since extracted features were unstable toward different face position transitions and rotations, lighting changes, etc. To combat this, geometric methods were used, in which key areas (lip area, eye area) [6] or key points (lips, eyes, eyebrows) [7] were extracted. To classify received face embeddings statistical classification methods [8] (linear discriminant analysis) or machine learning algorithms [3], [9] (hidden Markov model, support vector machine) were used.

Complexity and high variability of human facial expressions negatively affect the quality of intelligent emotion recognition systems based on single and static source of information, which is an image. To solve this problem, multimodal approaches have appeared that used additional data sources: electroencephalogram signals [10], speech [11], and others. Also, dynamic approaches [9] have appeared, in which a sequence of images were analyzed to capture the time component of information.

After artificial neural networks breakthrough, especially CNN development, the neural approach has become the main one for creating intelligent computer vision systems. As a result, convolutional neural networks began to be used for automatic emotion recognition [12]. Multi-headed neural networks were used as an implementation of a multimodal approach, and recurrent neural networks [13] and 4-d convolutional neural networks [14] were used to implement a dynamic approach.

Although deep learning models outperform classical methods in both feature extraction and its further classification, the computational complexity of neural networks makes it

978-1-6654-3941-1 @2021 IEEE

impossible to run full-size models on low performance hardware in real time. Thus, all reviewed methods propose either neural network based emotion recognition system, which is not suitable for real time inference on Raspberry Pi, or multistage classical computer vision system, which is hard to scale due to its heterogeneity. We propose scalable high accuracy neural network system for emotion recognition, which can be deployed on single board computer for real time inference.

III. Model development

Creation of described recognition system requires solving three main subtasks – developing a model for detecting a human face, developing a model for classifying emotions, and optimizing the resulting models for subsequent deployment on a single-board computer.

A. Detection model

Object detection is a computer vision task, which consists in identifying and localizing objects of a certain class, in our case, human faces. Localization is the task of finding the coordinates of the smallest rectangle that completely covers the object [15]. Over the past decade, impressive results have been obtained in solving the problem of detecting a human face. Detection models have evolved from a cascade of standard computer vision algorithms to end-to-end fully differentiable models [16]. Such a leap was largely due to the increase in computer performance, the development of GPU technologies and the reduction in the cost of computer memory. The latest solutions [17] require computing power of GPUs and large amounts of memory. On the contrary, to create a portable system running on a low-performance Raspberry Pi it is necessary to use a lightweight model. To satisfy such constraints, we chose YOLOv1 [18] convolutional neural network as baseline model. This model works faster than its counterparts, since it makes predictions by passing the entire image through single convolutional network, instead of separately processing supposed regions of interest. But this model still has too many trainable parameters, so it needs to be modified. To do this, we reduced the number of model layers, and replaced the fully connected layer with single convolution, which led to model weight reduction by several times. The resulting architecture of the detection model (Table 1) is typical for convolutional networks: N convolutional blocks followed by a head, in this case, detection layer A convolutional block is a sequence of two-dimensional convolution, batch normalization, activation, and pulling operations applied one after another.

Detection layer is a key component of the model. Input image is conventionally divided into a grid of cells, each of which is responsible for predicting whether it contains center of any object. Every spatial position of output tensor is associated with prediction for one cell. The prediction contains coordinates of the bounding box and score that shows how confident model is that the cell contains center of the object. Value of the confidence score should be close to the Jaccard coefficient calculated between the ground truth and predicted bounding boxes. It makes sense to predict several bounding boxes for each cell. Then, according to the confidence score for each rectangle, one box is selected as responsible for the prediction. This helps to overcome gradient instability caused by observations of different sizes, as different bounding boxes adjust to objects of various sizes.

Loss function (1) is a weighted sum of penalties for differences between ground truth labels and predictions. Only bounding boxes responsible for the prediction are penalized for coordinates regression task. More specifically, if i-th cell contains center of object and j-th bounding box predicted in this cell has the highest IoU with ground truth box, then indicator multiplier in regression terms of loss will equal to 1, otherwise to 0. Confidence score error is computed for both responsible and not responsible boxes with corresponding coefficients.

TABLE I. Detection model architecture

Layer	Type	#Filters	Size	Input	Output
1	Conv2D + BatchNorm + ReLU	8	3x3	288x288x3	288x288x8
	Conv2D + BatchNorm + ReLU	8	3x3	288x288x8	288x288x8
	MaxPool		2x2	288x288x8	144x144x8
2	Conv2D + BatchNorm + ReLU	16	3x3	144x144x8	144x144x16
	Conv2D + BatchNorm + ReLU	16	3x3	144x144x16	144x144x16
	MaxPool		2x2	144x144x16	72x72x16
3	Conv2D + BatchNorm + ReLU	32	3x3	72x72x16	72x72x32
	Conv2D + BatchNorm + ReLU	32	3x3	72x72x32	72x72x32
	MaxPool		2x2	72x72x32	36x36x32
4	Conv2D + BatchNorm + ReLU	64	3x3	36x36x32	36x36x64
	Conv2D + BatchNorm + ReLU	64	3x3	36x36x64	36x36x64
	MaxPool		2x2	36x36x64	18x18x64
5	Conv2D + BatchNorm + ReLU	128	3x3	18x18x64	18x18x128
	Conv2D + BatchNorm + ReLU	128	3x3	18x18x128	18x18x128
	MaxPool		2x2	18x18x128	9x9x128
6	Conv2D + BatchNorm + ReLU	192	3x3	9x9x128	9x9x192
	Conv2D + BatchNorm + ReLU	192	3x3	9x9x192	9x9x192
	Conv2D + BatchNorm + ReLU	192	3x3	9x9x192	9x9x192
	Conv2D + BatchNorm + ReLU	192	3x3	9x9x192	9x9x192
7	Conv2D	11	3x3	9x9x192	9x9x11

Result of any deep learning task strongly depends on amount and quality of training data. To obtain a high-quality model, we assemble dataset which contains about five thousand images of human faces with a high degree of variability in age and ethnic groups of captured faces, as well as in scenarios in which they are presented. To do this, several publicly available datasets [19], [20] were combined, manually and automatically filtered.

B. Classification model

After a person's face is localized, the task of emotion recognition is reduced to the task of facial expression classification. Task of emotion classification is rather complex, as categories are difficult to separate. Specifically, for the popular dataset FER2013 [21], in which 7 emotions are presented, accuracy that human achieves is only 65%. That is why massive convolutional neural networks [22] or complex models with recurrent connections [14] are used for emotion classification task. Thus, to develop a high quality light-weighted emotion classifier it is necessary to employ specially designed model architecture.

The final model (Table 2), that we propose for emotion classification problem, has a typical structure for convolutional networks, but uses techniques presented in the Xception model [23] to reduce the number of trained parameters and maintain the quality of the model.

To propagate important for expression recognition local features residual connections [23] were used. Another technique we used is depthwise separable convolution [24]. This type of convolution includes two sequential operations – pointwise convolution and depthwise convolution. The first is a convolution with a 1x1 kernel size and the second is a convolution with a 3x3 spatial kernel size, but with only one channel in depth. A standard convolutional layer simultaneously processes both inter-channel information (since the convolution is applied to all channels at once), and spatial information (pixel values within one channel). In contrast, depthwise separable convolution embodies assumption that spatial and channel information can be processed separately.

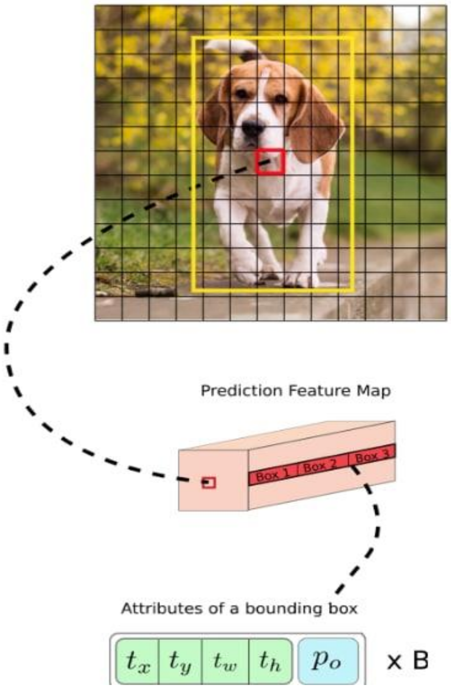

Fig. 1. Cell prediction [25]

Thus, use of such a type of convolutions grants number of trainable parameters reduction with no information loss.

To train the model, we prepare a dataset [21], [26], containing about 35000 images for 4 categories of emotions – neutral, angry, surprised, happy.

Loss function [18]:

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] +$$

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [\left(\sqrt{\omega_i} - \sqrt{\widehat{\omega}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \hat{C}_i\right)^2 + \qquad (1)$$

$$\lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} \left(C_i - \hat{C}_i\right)^2$$

C. Model optimization

Even though the recognition models were specially designed to perform cost-efficient inference, computations are still too complex to run system on Raspberry Pi in real-time mode. The complexity of model is mainly characterized by the number of its trainable parameters, which in the neural network can vary from hundreds of thousands to several million. Thus, to optimize model inference the size of trainable parameters should be addressed. Appropriate technique for doing such a type of optimization is model quantization [28], [29].

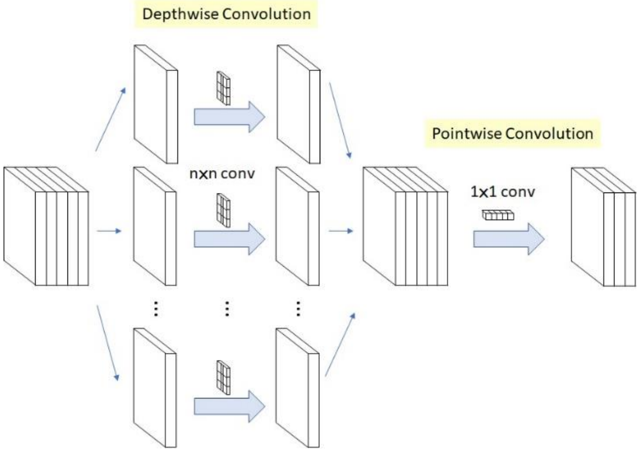

Fig. 2. Depthwise separable convolution layer [27]

Quantization is the process of reducing the number of bits that represent a number while maintaining the highest possible precision. That is, from 32-bit floating point arithmetic (float32) number is converted to more primitive integer formats for representing numbers. Although it is possible to convert to int32 or even to int4, optimal way is to convert to 8-bit numbers (int8). The advantage of using quantization is a significant increase in the speed of integer computations in comparison with floating point computations, as well as a multiple reduction in the amount of memory consumed by model.

Quantization of weights – the transition from float to int – occurs according to an expression parameterized by two values scale and offset:

$$x_{int} = x_{float} / x_{scale} + x_{offset} \tag{2}$$

The basic approach to quantizing a neural network is to train the model using full arithmetic and then just quantize weights. Although this approach has all the described advantages, it can lead to a deterioration in the quality of the model due to loss of information stored in the weights before quantization.

To combat the negative effects of quantization we use several special techniques:

1. Quantization aware training, which makes fake quantization during training allowing for higher accuracy. Specifically, during training all calculations are done in floating point, with modules modelling quantization effect by clamping and rounding to simulate int8 arithmetic. Backpropagation thereby can be done to find optimal parameters with regard to further quantization.
2. Post training calibration, which allows to determine optimal quantization parameters. In particular, we collect small dataset containing images similar to ones the model will operate with and propagate it through trained model. This made it possible to estimate the range of values of model activations and find parameters for precise quantization.

Also, activations were fused into the preceding layer where possible to yield model accuracy after quantization.

After that, models were traced [30] to gain extra execution performance. This converting allowed using just in time

TABLE II. CLASSIFICATION MODEL ARCHITECTURE

Layer	Type		#Filters	Size	Input		Output	
1	Conv2D		8	3x3	64x64x1		32x32x8	
	BatchNorm							
	ReLU							
2	Conv2D		8	3x3	32x32x8		16x16x8	
	BatchNorm							
	ReLU							
3	DepthSepConv	Conv2D	16	3x3, 1x1	1x1	16x16x8	16x16x16	8x8x16
	BatchNorm	BatchNorm						
	ReLU							
	DepthSepConv		16	3x3, 1x1	16x16x16		16x16x16	
	BatchNorm							
	MaxPool			3x3	16x16x16		8x8x16	
4	DepthSepConv	Conv2D	32	3x3, 1x1	1x1	8x8x16	8x8x32	4x4x32
	BatchNorm	BatchNorm						
	ReLU							
	DepthSepConv		32	3x3, 1x1	8x8x32		8x8x32	
	BatchNorm							
	MaxPool			3x3	8x8x32		4x4x32	
5	DepthSepConv	Conv2D	64	3x3, 1x1	3x3	4x4x32	4x4x64	2x2x64
	BatchNorm	BatchNorm						
	ReLU							
	DepthSepConv		64	3x3, 1x1	4x4x64		4x4x64	
	BatchNorm							
	MaxPool			3x3	4x4x64		2x2x64	
6	AdaptiveAvgPool			2x2	2x2x64		1x1x16	
7	Linear				16		4	

compiler for model inference rather than running it via interpreter.

IV. Results

The system for recognizing 4 basic emotions has been created, which is able to work on a single-board computer in real-time. The resulting composite model takes up only 6 MB and is capable of processing more than 4 frames per second on the Raspberry Pi 3.

TABLE III. RESOURCE INTENSITY OF MODELS

	Processing time per frame (ms)	**Model size (MB)**
Detection model	182	4.95
Classification model	56	1.1
Composite model	238	6.05

The obtained models show satisfactory results in proper tasks. The Jaccard similarity coefficient for the detection model is 0.86, and the accuracy for the classification model is 0.83. Model architectures and weights, as well as scripts for training and inference model, are available at GitHub [31].

Among all considered emotions only "anger" emotion is difficult for processing by the obtained model since it has a wide variability of external manifestations.

The obtained solution is also cost-effective, as it is undemanding to hardware resources. A single-board Raspberry Pi computer and web camera are all tools necessary for full system functioning. Taking it into consideration, the system is easily integrated into any portable device, such as robot.

V. Conclusions

The created emotion recognition system has a unique set of properties. Unlike multi-stage solutions that use classical algorithms of computer vision and machine learning, the neural network solution allows easily scale the system to improve the quality of recognition, depending on the available computing resources. The architectural solutions used in the development of neural networks and advanced optimization techniques, such as quantization, made it possible to make the system lightweight and fast. Comparing to the existing TinyYOLO and MiniXception models used as baseline models we managed to achieve more than a tenfold reduction in the amount of memory used and a multiple increase in the frame processing speed. At the same time, the system demonstrates a high quality of predictions, comparable to heavy full-size models. The system is easily embedded into any IoT system, as it requires a minimum of hardware tools for its operation, which makes the solution also cost-effective.

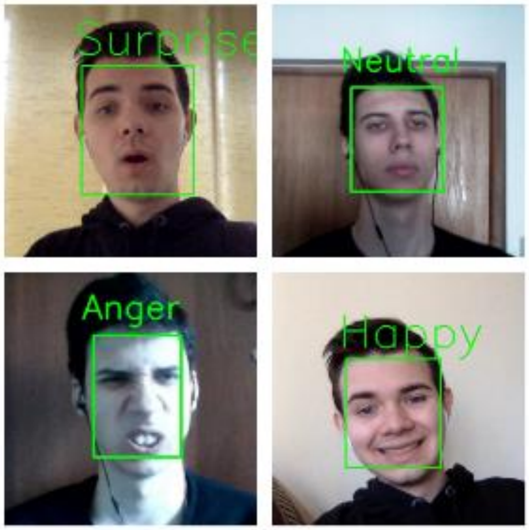

Fig. 3. Some detection and classification results

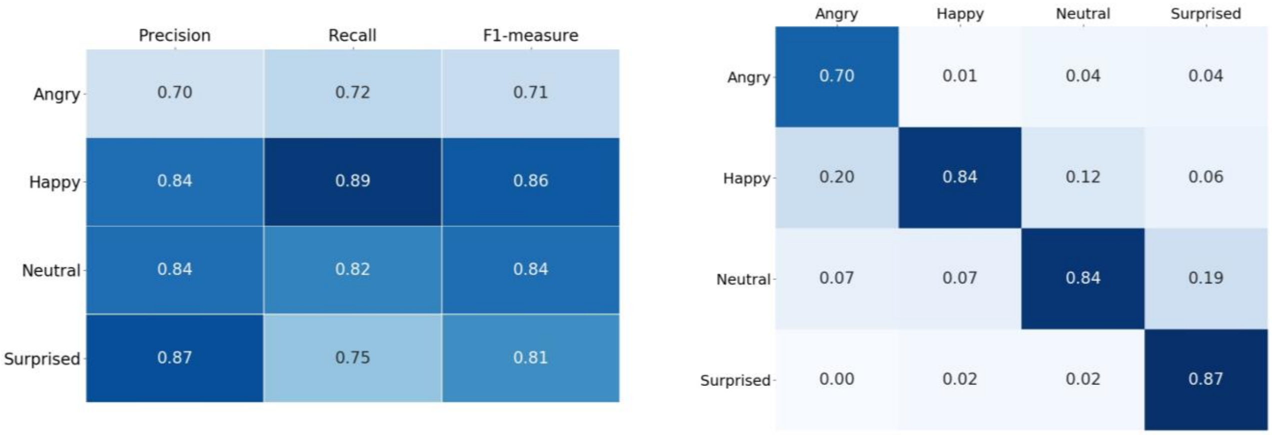

Fig. 4. Experimental results of our classification model

REFERENCES

[1] P. Ekman and W. V Friesen, “Constants across cultures in the face and emotion,” J. Pers. Soc. Psychol., vol. 17, no. 2, 1971, pp. 124–129

[2] L. Asiedu, A. Adebanji, F. Oduro, and F. Mettle, “Statistical Assessment of PCA/SVD and FFT-PCA/SVD on Variable Facial Expressions,” Br. J. Math. Comput. Sci., vol. 12, no. 6, 2016.

[3] B. T. Lau, “Portable real time emotion detection system for the disabled,” Expert Syst. Appl., vol. 37, no. 9, 2010, pp. 6561–6566.

[4] P. L. V. D. M. Sasikumar and others, “A neural network based facial expression analysis using Gabor wavelets,” Word Acad. Sci. Eng. Technol., 2008.

[5] M. Satiyan and R. Nagarajan, “Recognition of facial expression using haar-like feature extraction method,” 2010.

[6] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, “Interactive facial feature localization,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2012, vol. 7574 LNCS, no. PART 3, pp. 679–692.

[7] S. Arca, P. Campadelli, and R. Lanzarotti, “A face recognition system based on automatically determined facial fiducial points,” Pattern Recognit., vol. 39, no. 3, 2006.

[8] F. Z. Jahromy, A. Bajoulvand, and M. R. Daliri, “Statistical algorithms for emotion classification via functional connectivity,” J. Integr. Neurosci., vol. 18, no. 3, pp. 293–297, 2019.

[9] R. A. Patil, V. Sahula, and A. S. Mandal, “Facial expression recognition in image sequences using active shape model and SVM,” 2011.

[10] A. Hassouneh, A. M. Mutawa, and M. Murugappan, “Development of a Real-Time Emotion Recognition System Using Facial Expressions and EEG based on machine learning and deep neural network methods,” Informatics Med. Unlocked, vol. 20, 2020, p. 100372.

[11] Y. Wang, X. Yang, and J. Zou, “Research of Emotion Recognition Based on Speech and Facial Expression,” TELKOMNIKA Indones. J. Electr. Eng., 2013.

[12] N. Mehendale, “Facial emotion recognition using convolutional neural networks (FERC),” SN Appl. Sci., vol. 2, no. 3, 2020, pp. 1–8.

[13] Y. Fan, X. Lu, D. Li, and Y. Liu, “Video-Based emotion recognition using CNN-RNN and C3D hybrid networks,” in ICMI 2016 - Proceedings of the 18th ACM International Conference on Multimodal Interaction, 2016, pp. 445–450.

[14] X. Ouyang et al., “Audio-visual emotion recognition using deep transfer learning and multiple temporal models,” 2017.

[15] A. D. Rhodes, M. H. Quinn, and M. Mitchell, “Fast on-line kernel density estimation for active object localization,” 2017.

[16] Z. Zou, Z. Shi, Y. Guo, and J. Ye, “Object detection in 20 years: A survey,” arXiv Prepr. arXiv1905.05055, 2019.

[17] M. Tan, R. Pang, and Q. V. Le, “EfficientDet: Scalable and efficient object detection,” 2020.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016.

[19] B. Yang, J. Yan, Z. Lei, and S. Z. Li, “Fine-grained evaluation on face detection in the wild,” in 2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, FG 2015, 2015, pp. 1–7.

[20] S. Yang, P. Luo, C. C. Loy, and X. Tang, “WIDER FACE: A face detection benchmark,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-Decem, pp. 5525–5533.

[21] I. J. Goodfellow et al., “Challenges in representation learning: A report on three machine learning contests,” Neural Networks, vol. 64, pp. 59–63, 2015.

[22] B. Hasani and M. H. Mahoor, “Facial Expression Recognition Using Enhanced Deep 3D Convolutional Neural Networks,” in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017, vol. 2017-July, pp. 30–40.

[23] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-Decem, pp. 770–778.

[24] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017, vol. 2017-Janua, pp. 1251–1258 .

[25] A. Kathuri, “How to implement a YOLO (v3) object detector from scratch in PyTorch: Part 1,” PaperspaceBlog, Apr. 16, 2018. https://blog.paperspace.com/how-to-implement-a-yolo-object-detector-in-pytorch/ (accessed Mar. 14, 2020).

[26] D. Lundqvist, A. Flykt, and A. Ohman, “The Karolinska directed emotional faces (KDEF),” CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet. 1998.

[27] S.-H. Tsang, “Review: Xception — With Depthwise Separable Convolution, Better Than Inception-v3 (Image Classification),” Towards Data Science, Sep. 25, 2018. https://towardsdatascience.com/review-xception-with-depthwise-separable-convolution-better-than-inception-v3-image-dc967dd42568 (accessed Mar. 18, 2020).

[28] Y. Guo, “A survey on methods and theories of quantized neural networks,” arXiv Prepr. arXiv1808.04752, 2018.

[29] K. Paupamah, S. James, and R. Klein, “Quantisation and pruning for neural network compression and regularisation,” 2020.

[30] Torch Contributors, “Torchscript,” Pytorch, 2019. https://pytorch.org/docs/stable/jit.html (accessed May 02, 2020).

[31] S. Mikhaylevskiy, V. Chernyavskiy, and V. Pavlishen, “Emotion-Recognition-PRJCT2019,” GitHub, 2020. https://github.com/mixst99/Emotion-Recognition-PRJCT2019 (accessed Apr. 10, 2020).

[32] O. Arriaga, M. Valdenegro-Toro, and P. G. Plöger, “Real-time convolutional neural networks for emotion and gender classification,” 2019.

[33] P. Priyanka and T. R. R. Kumar, “Real-time Facial Expression Recognition System using Raspberry Pi,” 2017, doi: 10.22161/ijaers/nctet.2017.ece.2.

[34] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” 2017, doi: 10.1109/CVPR.2017.690.

[35] Y. Zhang and Q. Ji, “Facial expression understanding in image sequences using dynamic and active visual information fusion,” in Proceedings of the IEEE International Conference on Computer Vision, 2003, vol. 2, pp. 1297–1304.