Science Journal of Circuits, Systems and Signal Processing
2018; 7(4): 81-86
http://www.sciencepublishinggroup.com/j/cssp
doi: 10.11648/j.cssp.20180704.11
ISSN: 2326-9065 (Print); ISSN: 2326-9073 (Online)

Real-Time Image Compression System Using an Embedded Board

Aurelle Tchagna Kouanou[1, *], Daniel Tchiotsop[2], Theophile Fonzin Fozin[1], Bayangmbe Mounmo[3], René Tchinda[4]

[1]Research Unity of Condensed Matter, Electronics and Signal Processing, Department of Physics, Faculty of Science, University of Dschang, Dschang, Cameroon

[2]Research Unity of Automatic and Applied Informatic, IUT-FV of Bandjoun, University of Dschang-Cameroun, Cameroun, Bandjoun

[3]National Advanced School of Posts, Telecommunications and Information Communication and Technologies (SUP'PTIC), Yaounde, Cameroon

[4]Research Unity of Engineering Industrial Systems and Environments, University of Dschang, Cameroun, Bandjoun

Email address:

tkaurelle@gmail.com (A. T. Kouanou)

[*]Corresponding author

To cite this article:

Aurelle Tchagna Kouanou, Daniel Tchiotsop, Theophile Fonzin Fozin, Bayangmbe Mounmo, René Tchinda. Real-Time Image Compression System Using an Embedded Board. *Science Journal of Circuits, Systems and Signal Processing*. Vol. 7, No. 4, 2018, pp. 81-86.
doi: 10.11648/j.cssp.20180704.11

Received: January 26, 2019; **Accepted**: March 7, 2019; **Published**: March 27, 2019

Abstract: Digital image processing is the use of computer algorithms to improve image quality, to extract or add information. Image compression is a part of image processing and is used to reduce the quantity of data to store. This paper presents the implementation of image compression operations on low cost embedded systems (Raspberry Pi 3, Arduino Uno R3). Motivated by the work of Tchagna et al. (DOI: 10.5815/ijigsp.2018.11.05), we proposed within this paper a real-time implementation of the compression algorithm on embedded boards. Our investigation in this paper is to make a real-time compression system able to capture an image, apply compression algorithm and save compression image on an SD card (for Arduino or Raspberry) or sent directly compressed image to cloud. Compression system with a Raspberry Pi basically used a webcam USB camera to capture the images, the compression function based on python language, and a function to store compressed image to an SD card or to Cloud. The compression system with Arduino used an SD card where the image to be compressed are stored, an external SRAM chip, and an Ethernet shield. The proposed hardware system can decompress the image. In opposition to the approach adopted in the literature, all the results presented within this work use the vector quantization. Eight images have been used to evaluate and compared the compression time for each board according to codebook size used during vector quantization step. Based on our results, we remark that compression and decompression time using Raspberry Pi is lower than compression and decompression time using Arduino. Raspberry Pi offers many possibilities and its processor is bigger than Arduino processor. This justifies the obtained compression and decompression time using Raspberry Pi compared to those with Arduino.

Keywords: Arduino Uno, Raspberry Pi, Image Compression, Vector Quantization, Embedded System

1. Introduction

Image compression is a form of signal processing, and a type of data compression applied to an image to reduce its cost for storage or transmission. Image compression is an extremely important part of modern computing. By having the ability to compress images from their original size, the required storage space in computer memory space of images can be reduced [1]. In modern sciences and technologies, images gain much broader scopes due to the ever-growing importance of scientific visualization (of often large-scale complex scientific/experimental data) like microarray data in genetic research, or real-time multi-asset portfolio trading in finance [2]. To operationalize the image compression algorithm, they are embedded in low-cost microcomputers. The specifications of these micro-computers (clock

frequency, memory size, kind, and a number of peripheral devices such as timers, analog-to-digital converters (ADC), etc.) are set according to the number and type of input/output lines and the complexity of the algorithms running in them [3]. Nowadays, Arduino and Raspberry are used in many digital signal/image/video applications due to the low cost of these equipment's to perform a processing [2, 4-9] or use for transmission [10-12]. Arduino Uno and Raspberry Pi come with an easy development environment; their size offers many possibilities of use in electronic, computer science and telecommunications. There exist two types of image compression, lossless and lossy compression. In lossless compression, the reconstruction image is identical to the original and compression rates are very low; and in lossy compression, there is a loss of data but the compression ratio is very high. Tchagna et al., used lossy compression coupled with machine learning algorithm to vector quantization in order to improve this step [1]. They concluded that the lossy compression performs with discrete Walsh transforms, vector quantization with k-mean and splitting method and Huffman coding was the method that gave the best compromise between image quality and a performance parameter. Thus, in this paper, we look for how we can embed this algorithm method into Raspberry Pi and Arduino Uno.

In this paper, we will focus specifically on image compression into an embedded board, along with Raspberry Pi and Arduino Uno. An architectural workflow describes the optimal algorithm and method reported in the literature. We will present a flow chart performing the steps of acquisition of image, compression and store or sharing. We describe the importance to use a low-cost embedded board to perform the image compression. Two main hardware architectures are proposed. The one is based on Raspberry Pi and the other is based on Arduino Uno. The two proposed architectures hardware will be compared in term of compression and decompression time according to codebook size used during the vector quantization step. The rest of this paper is organized like this: the state of art and the presentation of several works in the area are contained in section 2. In Section 3, the different parts of the proposed hardware architecture are explained in detail. Section 4 outlines the results that were carried out and discussion. Section 5 presents the conclusions and future work.

2. State of the Art

Arduino and Raspberry can be considered in another case as a small computer without a screen, keyboard and mouse. There is currently a growing interest in interconnecting these small computers to increase their interoperability and control through internal or external networks [3]. Arduino and Raspberry Pi are the low-cost device and can be used to embedded many algorithms, automatic and monitoring the systems. In an embedded market study about integrating IoT and Advanced Technology Design, Arduino was the OSHW platform used most in current embedded designs (5.6%) or considered and being considered for use in the near future (17%), followed by Raspberry Pi (4.2% and 16%) and Beagle Bone (3.4% and 10%) [3].

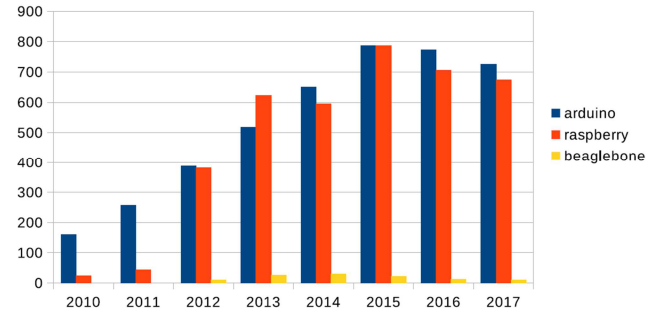

Figure 1. *Google trends for Arduino, Raspberry Pi and Beaglebone boards in the last years [3].*

Figure 1 shows us the google trends for Arduino, Raspberry Pi and Beaglebone boards in the last years where Arduino boards show slightly higher results than Raspberry Pi. This figure justifies why our work deal with Arduino and Raspberry Pi. These platforms have been positively included in the electrical and electronic engineering studies with a huge benefit to both curriculum and students [13]. Many papers have been published concerning the use of a minicomputer built on platform Arduino, Raspberry Pi, Node MCU, for the purposes of image compression and decompression or image/video processing. It's the case of Sieczkowski and Sondej that present in 2017 an analysis of the use of a single-board computer Raspberry PI for video acquisition and transmission [5]. They focused on requirements necessary for the recorded image to be used for face analysis to identify facial expressions and microexpressions [5]. They have given the effect of resolution and frame rate on the quality of the recorded video stream and calculated the compression ratio and PSNR [5]. However, the compression ratio and PSNR were very low. An average of 50% and 17 dB for compression ratio and PSNR respectively. The obtained PSNR prove that the image quality was no good. Molina-Cantero et al., proposed in 2018 novel software architecture based on open-software hardware platforms which allow programmers to create data streams from input channels and easily implement filters and frequency analysis objects [3]. The software allows programmers to implement basic signal processing applications easily in open-source hardware platforms like Arduino Genuino, one of the most popular, well-known, supported and inexpensive platforms [3]. The authors did not make any test on Arduino 32-bit. Al-Ani presents a real-time image compression based on Raspberry Pi using Discrete Wavelet Transform and scalar quantization [14]. But he hasn't evaluated the time for compression and decompression image according to its image size. In 2017, Lazar et al. proposed an image compression and decompression based on JPEG algorithm on the platform Arduino. They used two Arduino platforms': 8-bit platform and 32-bit platform and demonstrated that performance of the 32-bit platform was much higher [6]. The authors present a real-time application of Raspberry Pi in compression of images [7] and Kumar and

Murthy [15] proposed a novel video compression technique since by saving the bandwidth of the wireless sensor networks (WSN). Their proposed video compression algorithms are implemented on the Raspberry Pi platform using python language and communicated wirelessly to the base stations (laptop) using low power Zigbee radio trans-receivers [15]. However, their video compression technique is based on JPEG method. In these four previous works, the authors used JPEG algorithm which relies on scalar quantization and is not suitable for some kind of images like biomedical images. Also, the quality of the decompressed image with JPEG method is not good. In 2018, Tchagna et al. proposed a compression system for biomedical image using a machine learning and orthogonal transform (Walsh transform and Chebyshev transform) [1]. They coupled k-mean clustering with splitting method to optimize the vector quantization step. The results in term of compression ratio and peak signal to noise ratio, for example, were very impressionable. Their work was limited to simulation on a computer.

To the best of our knowledge, none of the existing researches presents an image compression into Raspberry Pi or Arduino Uno using vector quantization. To overcome this drawback, we based upon distortion factor considerations introduced by Shannon [16, 17] where the best performance can be obtained using vectors instead of scalars. We proposed in this paper to embed a compression method based on vector quantization by using two kinds of embedded system: Arduino Uno R3 and Raspberry Pi 3 B. The aim is to embed the compression algorithm proposed [1] in this two electronic boards. This is to observe practically the comportment of a compression system in a real-time.

3. Materials and Methods

3.1. Materials

We have implemented our work in embedded systems, namely Raspberry Pi 3 B and Arduino Uno R3. The Raspberry Pi 3 B is chosen to perform complex operations thanks to its multi-core processor. The main difference between the versions of the Raspberry Pi computer is the SoC (System-on-Chip) used. This system depends on the computer version includes a single-core ARM11 processor, a quad-core ARM Cortex-A7 processor or a quad-core ARM Cortex-A53 processor [5]. Raspberry Pi 3 intended for image acquisition and compression, a key role is played by video camera and the interface to which it is connected. Currently, CSI (Camera Serial Interface) and USB are the most commonly used. Type of camera used also depends to a large extent on the intended use of the target device [5]. This paper focuses on the acquisition of image to compress and store or share. For this reason, specific devices regarding the quality of the captured image must be met. Arduino is the most affordable budget solution, implemented on the popular AVR microcontrollers and allows using C/C++ and Assembler languages the most requested for DSP [4]. The Arduino Uno R3 is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller and it has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs) [18]. Figure 2 presents us the Raspberry Pi, Arduino Uno boards and another device that we used and Table 1 gives us the hardware specifications of each.

Table 1. *System hardware specifications for Arduino Uno R3 and Raspberry Pi 3 B [18-20].*

Arduino System	Arduino Uno R3	Processor	ATmega328P
		CPU Speed	16 MHz resonator
		Digital I/O	14
		Analogue inputs	6
		PWM outputs	6
	Micro External SD Card	Capacity	8Go
	External SRAM (23LC1024)	Capacity	128 kB
	Ethernet Shield		
Raspberry System	Raspberry Pi 3	Processor	Broadcom BCM2837 64bit Quad Core
		CPU Speed	1.2 GHz
		Digital I/O	40pin extended GPIO
	Webcam Camera	Quality	5 M Pixels

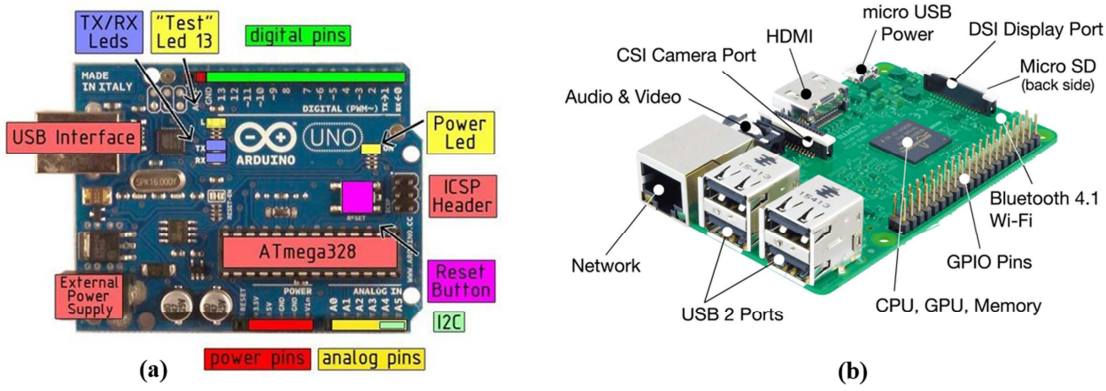

Figure 2. *Arduino Uno R3 and Raspberry Pi 3 nomenclature [18-20].*

3.2. Methods

This part deals with the development of software and hardware for Raspberry Pi 3 B and Arduino Uno R3 which enable to compress and decompress an image. On the one hand, we extended the Arduino Uno board with an external SRAM chip 128 kB, SD slot and Ethernet Shield. Compressed and decompressed data image are stored on an SD card 8 GB into SD slot. The compression algorithm performed [1] is uploaded by using the Arduino IDE into Arduino microcontroller and the real-time compression can start. Ethernet shield is used in the case where we want to share compressed algorithm or store it to a cloud. Figure 3 (a) shown the basic block diagram for image compression using Arduino.

In the other hand, we implement the compression method based on Walsh transform, vector quantization and Huffman encoding on raspberry Pi 3 processor in Open CV platform using python as its programming language. Open CV provides many modules for programming of algorithm (like k-means algorithm). Figure 3 (b) presents the process of image compression on Raspberry Pi 3. The image captured by a USB camera has to be compressed in real-time. The images captured by USB camera will have raw data with more storage space hence it is important to compress the image using the method proposed in [1]. The compression ratio and quality level are controlled by codebook size used during vector quantization step.

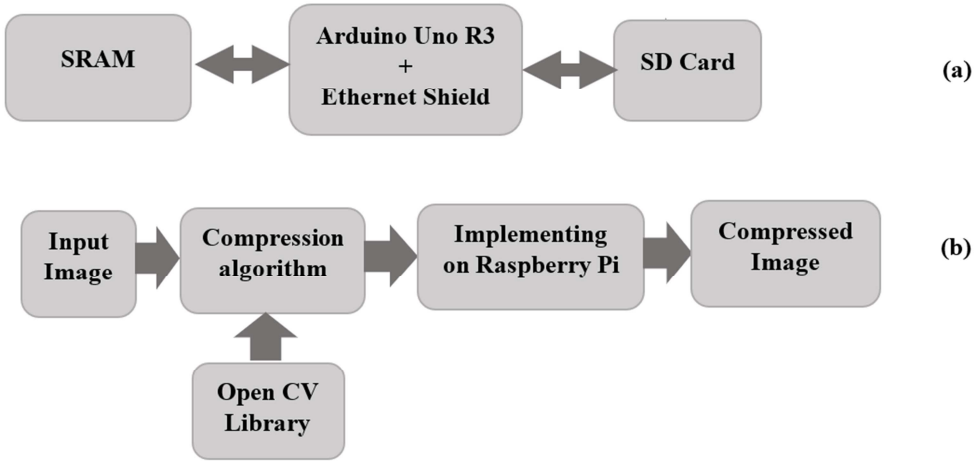

Figure 3. *Basic block diagram for image compression. (a) on Arduino Uno R3; (b) on Raspberry Pi 3 B.*

4. Results

There were eight tests on Arduino platform and eight tests on Raspberry Pi platform carried out (with image size 256X256). Figure 4 shows the real implementation of our both prototypes Arduino and Raspberry. On Arduino board, we have inserted directly the SD card in SD slot of Ethernet Shield.

Figure 4. *Real presentation of Raspberry Pi with USB camera and Arduino Uno with Ethernet Shield and SD card.*

Table 2 shows the average compression and decompression time with Raspberry Pi and Arduino Uno according to the codebook size used during the vector quantization step. For a better visualization, average times of table 2, shown in the graphs displayed in Figure 5, show that the compression and decompression time increase when the

codebook size increase. This remark is the same observed in Ref [1] and when the codebook size equal to 256 the decompressed image is almost the same as the original image.

Table 2. *Average compression and decompression time according to the codebook size.*

Embedded Boards	Average Compression Time	Average Decompression Time	Codebook Size
Raspberry Pi 3 B	40.55	77.12	32
	46.72	96.44	64
	64.01	110.59	128
	86.11	151.09	256
Arduino Uno R3	145.55	279.19	32
	193.72	350.97	64
	255.01	458.56	128
	309.11	614.42	256

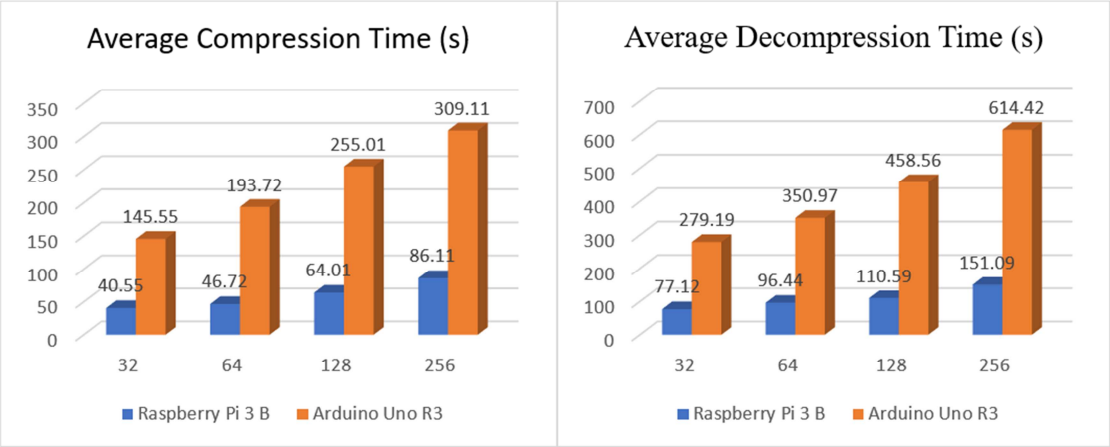

Figure 5. *Average compression and decompression graphs time according to codebook size.*

Regarding the graphs of Figure 5, we remark that the decompression time is always bigger than compression time. Moreover, in Figure 5, when the codebook size increase, the decompression and compression time increase also. This is because the time to generate the codebook using during vector quantization step becomes more important. The compression/decompression time using Raspberry Pi is less than compression/decompression time using Arduino. This can be explained thanks to table 1 where we observe that the frequency processor of Arduino and Raspberry Pi is 16MHz and 1.2 GHz respectively.

In this paper, we performed the image compression/ decompression by using two low cost embedded boards (Arduino Uno R3 and Raspberry Pi 3 B). These embedded boards are available almost in all country due to its low cost. However, Arduino board is less expensive than Raspberry Pi board. Thanks to the size of these both boards, they are easy to manipulate and their programming also. The first difference between our work and literature work is the use of vector quantization and Walsh transform. In many related works [6, 7, 14, 15] the authors deal with JPEG compression method and most of the available works provide information about the principles and algorithms used in JPEG with Arduino or Raspberry Pi.

5. Conclusion

Image compression has a wide range of applications and they exist many techniques to perform the compression, one of the effective techniques is the use of discrete Walsh transform, vector quantization and Huffam encoding. In this work, image compression is implemented using two embedded boards (Raspberry Pi 3 B and Arduino Uno R3). The obtained results in term of compression and decompression time show us that Raspberry Pi 3 give us a better result than Arduino Uno R3. Due to the low cost of these two embedded boards, they are suitable to develop an effective hardware system to manage the images in an institution like manage biomedical images in a hospital. In the future work, we are going to use the Spark framework and MapReduce programming into Raspberry Pi 3 in order to compresses many images at the same time as present in [21, 22].

References

[1] Tchagna KA, Tchiotsop D, Tchinda R, Tchapga CT, Kengnou Telem AN, Kengne R, A (2018) Machine Learning Algorithm for Biomedical Images Compression Using Orthogonal Transforms, I. J. Image, Graphics and Signal Processing 11, 38-53. DOI: 10.5815/ijigsp.2018.11.05.

[2] Kar S, Jana A, Chatterjee D, Mitra D, Banerjee S, Kundu D, Chosh S, Gupta SD (2015), Image Processing Based Customized Image Editor and Gesture Controlled Embedded Robot Coupled with Voice Control Features, (IJACSA) Int. J. of Advanced Computer Science and Applications 6 (11), 91-96.

[3] Molina-Cantero AJ, Castro-Garcia JA, Lebrato-Vasquez C, Gomez-Gonzalez IM and Merino-Monge M (2018) Real-Time Processing Library for Open-Source Hardware Biomedical Sensors, Sensors 18, 1033; doi:10.3390/s18041033.

[4] Vostrukhin A and Vakhtina E, Studying Digital Signal Processing on Arduino Based Platform (2016), Engineering for Rural Development, 236-241.

[5] Sieczkowski K and Sondej T (2017) Use of a Raspberry PI single-board computer for image acquisition and transmission, Measurement Automation Monitoring 63 (5), 180-182.

[6] Lazar J, Kostolanyova K, Bradac V (2017) Processing and image compression based on the platform Arduino, AIP Conference Proceedings 1863, 070025. http://dx.doi.org/10.1063/1.4992247.

[7] Sahitya S, Lokesha H, Sudha LK, Real Time Application of Raspberry Pi in Compression of Images, IEEE International Conference on Recent Trends in Electronics Information Communication Technology, May 20-21, 2016, India. 978-1-5090-0774-5/16/$31.00.

[8] Sushma G, Joseph M, Tabitha AR, Yokesh MBP (2015) Image Tracking Based Home Security Using Arduino Microcontroller, Int. J. of Innovative Research in Computer and Communication Engineering 3 (8), 117-122.

[9] Shilpashree KS, Lokesha H, Shivkumar H (2015), Implementation of Image Processing on Raspberry Pi, Int. J. of Advanced Research in Computer and Communication Engineering, 4 (6), 199-202. DOI 10.17148/IJARCCE.2015.4545.

[10] Yan Z, Che X, Walker J, Remote Image Sensing Platform Based on Arduino, IEEE 6th Computer Science and Electronic Engineering Conference, 25-26 Sept, 2014, Colchester. 10.1109/CEEC.2014.6958550.

[11] Tchapga Tchito C, Tchiotsop D, Fomethe A, NodeMCU in Patient's Data Transfer to IoT Platform (2018) Journal of Biomedical Engineering and Medical Imaging 5 (3), 9-18, DOI: 10.14738/jbemi.53.4358.

[12] Shah D and Haradi V (2016) IoT Based Biometrics Implementation on Raspberry Pi, Procedia Computer Science 79, 328 – 336. doi: 10.1016/j.procs.2016.03.043.

[13] Jamieson P and Herdtner J, More missing the Boat—Arduino, Raspberry Pi, and small prototyping boards and engineering education needs them. In Proceedings of the Frontiers in Education Conference (FIE), Washington, 21–24 Oct, 2015 1–6.

[14] Al-Ani MS, Hardware Implementation of a Real Time Image Compression (2017) IOSR Journal of Computer Engineering (IOSR-JCE) 19 (3), 6-13. DOI: 10.9790/0661-1903050613.

[15] Kumar Shiva NR and Venkatesh Murthy NK (2016), Design and Implementation of Novel Video Compression Technique Using Raspberry Pi, Int. J. of Advanced Research in Electrical, Electronics and Instrumentation Engineering 5(4), 3400-3404. DOI:10.15662/IJAREEIE.2016.0504175.

[16] Sayood K, Introduction to Data Compression. Morgan Kaufmann, San Francisco, 2006.

[17] Salomon D, A Concise Introduction to data compression, Springer-Verlag, London, 2008.

[18] Fitzgerald, Shiloh M, Igoe T, The Arduino Projects Book, Arduino LLC, Torino, 2012.

[19] Hughes JM, Arduino: A Technical Reference, O'Reilly Media, Sebastopol, 2016.

[20] Halfacree G, Raspberry Pi: Beginner's Guide, Raspberry Pi Trading Ltd, Cambridge, 2018.

[21] Tchagna KA, Tchiotsop D, Kengne R, Djoufack ZT, Ngo Mouelas AA, Tchinda R (2018) An Optimal Big Data Workflow for Biomedical Image Analysis, Informatic in Medicine Unlocked 11, 68–74. https://doi.org/10.1016/j.imu.2018.05.001.

[22] Kouanou AT, Tchiotsop D, Tchinda R, Tansaa ZD (2019) A Machine Learning Algorithm for Image Compression with application to Big Data Architecture: A Comparative Study. Br Biomed Bull Vol.7 No.1:316.