AEIPA: Docker-based system for Automated Evaluation of Image Processing Algorithms

Conference Paper · April 2017

DOI: 10.1109/CCECE.2017.7946720

3 authors, including:

Theodore Norvell
Memorial University of Newfoundland
47 PUBLICATIONS **193** CITATIONS

SEE PROFILE

AEIPA: Docker-based system for Automated Evaluation of Image Processing Algorithms

Acil Abdel Naby, Mohamed S. Shehata and Theodore S. Norvell
Electrical and Computer Engineering, Faculty of Engineering and Applied Science
Memorial University of Newfoundland
St. John's, Newfoundland, Canada
Email: {arian1, mshehata, theo}@mun.ca

***Abstract*—Evaluation of an image processing algorithm (IPA) task is tedious and full of complexity. A lot of effort is spent to compare a new IPA with benchmark IPAs. Comparing with benchmark IPAs requires either implementing them from scratch or a lot of configurations. Also, setting up datasets for the evaluation consumes additional effort. Therefore, the need for a system to overcome the former overhead is imperative. In this paper, a design of a novel automated evaluation system, AEIPA, is proposed. AEIPA allows automatic evaluation of different IPAs using different datasets regardless the programming language of the IPAs. Also, automatic reporting module is provided to compare different IPAs results. The proposed system applies openness principal to enrich AEIPA with IPAs (using plugin-based concepts in Docker containers), and datasets.**

I. Introduction

Image processing is a dynamic research area that is being continuously investigated. It is involved in many real-life applications, such as traffic monitoring [1], quality inspection [2] automobile parking [3], human identification [4], just to name a few. These applications add a lot of challenges which require a rapid evolution of IPAs.

However, too much effort is spent in evaluating the IPAs instead of focusing on IPAs enhancement. Typically, the Researches (who are the main users of AEIPA) have to go through all the following steps:

- Implementing all the related benchmark IPAs
- Setting up benchmark datasets to be ready for use
- Creating new datasets to test different challenges
- Evaluating each of the related benchmark IPAs individually
- Reporting the results which usually needs writing scripts/code manually

Although there are online repositories [5] that contain the implementation of the benchmark IPAs and benchmark datasets, researches get overwhelmed with configuring these benchmark IPAs and datasets to make them ready for use, and facing all the troubles of customizing them according to the working environment.

In this paper, AEIPA (Automated Evaluation of Image Processing Algorithms) design is presented to solve the overhead that was mentioned above. It allows automatic execution of IPAs, comparing them to already existing benchmark IPAs, and evaluation of the results. AEIPA is structured into three architectural modules: AEIPA Supplier, AEIPA Factory, and Docker Engine. AEIPA is isolated into a Docker container to allow agility and efficiency of the continuous integration and deployment activities. To the best of our knowledge, AEIPA (Automated Evaluation of Image Processing Algorithms) is the first system to solve these problems and facilitates the process of developing a new IPA.

The paper is arranged as follows: Section II depicts the background of the paper. Section III illustrates the proposed system. The conclusion and future work are in section IV.

II. II. BACKGROUND

General concepts are introduced in this section to be used in the rest of the paper.

A. Machines and Containers

Virtual Machines (VMs) contain the entire operating system. VMs suffer from an overhead in translating machine instructions from guest to host OS [6]. Also, VMs do not share libraries, and system files, etc. between guests and hosts OS.

Containers, on the other hand, are technologies that facilitate packing applications to be platform independent. These technologies allow wrapping the application dependencies, e.g. libraries and binaries, in a single package, which in turn simplifies and speeds up the deployment of the application. To put a finer point on it, imagine a developer implementing an application using Python 2.7. Developers may face incompatibility issues when developing in Python 2.7 while the production server has Python 3 installed. There are other dependencies that may generate problems when using different environment such as OS distributions, system libraries, security and storage policies, and network topologies.

Comparing Containers and VMs definitions, containers allow more agility to the applications. Also, the comparisons in [7-11] between containers and VMs show that containers have performance and storage improvements and reduced startup time. Hence, the proposed system adopts containers technologies.

B. Docker

Docker is a daemon that implements extra layers above Container adding Union File System capability and image management (image is package contains an application and its

978-1-5090-5538-8/17/$31.00 ©2017 IEEE

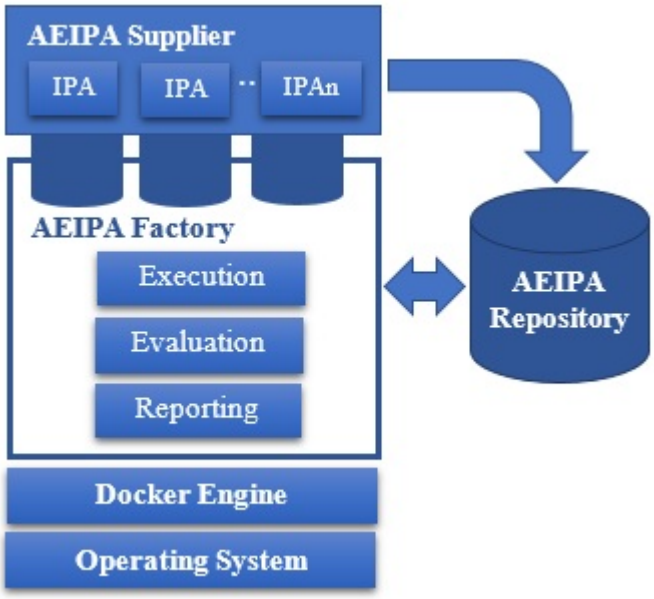

Fig. 1. AEIPA Architecture

dependences) to it. Also, it adds security and resource isolation by using cgroups and namespaces [12, 13]. Docker consists of: Docker Engine and Docker Hub. Docker Engine is the core technology that builds and runs Docker containers which are instances of Docker images. Docker Hub is a Cloud-based service that shares Docker images. It allows building, testing, and storing Docker images.

Docker makes installation a lot easier and unlimitedly replicable. It packages deployments into images without starting one up from scratch. The contents of a Docker image are defined online in Docker Hub. Hence, Docker images can be pushed/pulled to/from Docker Hub without cluttering the hosting machine with lots of files. In AEIPA, Docker container is used for all the mentioned advantages.

III. PROPOSED SYSTEM

AEIPA is a proposed automated evaluation system for IPAs. It will allow evaluating and comparing new IPAs with benchmark IPAs without the need for any customizations or configurations. Also, it hides the overhead required to deal with input datasets (such as images, videos) and live streaming from cameras. Moreover, automatic reporting of results is provided.

A. System Architecture

AEIPA is composed of three modules: AEIPA Supplier, AEIPA Factory, and Docker Engine as shown in Figure 1. AEIPA Supplier allows the users to provide new IPAs to AEIPA. AEIPA Factory is responsible for executing, evaluating and comparing different IPAs, a well as reporting the results with the least user intervention. AEIPA Repository is where all the results are stored. The proposed system is based on Docker Engine which wraps AEIPA with its dependencies, and the IPAs into Docker images to make them cross platform.

1) AEIPA Supplier: AEIPA Supplier plugs IPAs into AEIPA Factory in Docker image format. More precisely, user just needs to provide an IPA in plain source code, and AEIPA Supplier automatically converts the IPA into a Docker image that packages the code with the dependencies. Hence, the generated IPAs in Docker image can be executed in any environment. Also, users do not require prior knowledge of Docker because of the automatic conversion to Docker image. AEIPA Supplier stores all Docker images in AEIPA Repository for future use.

IPA can be written using any preferred language supported by either Docker Hub or directly by AEIPA, such as C/C++ (GCC), Java, PHP, Python, Hy (Hylang), Go (Golang), Node, Perl, Rails, Clojure, and Ruby. As computer vision researches are the targeted users of AEIPA, AEIPA Supplier converts IPA source code written Matlab and OpenCV as well.

2) AEIPA Factory: AEIPA Factory consists of three submodules: Execution, Evaluation, and Reporting. Execution deploys the users IPA in parallel with other benchmark IPAs that the user wants to compare with. Evaluation converts the visual output of the executed IPA(s) into quantitative measures, e.g. True Positive Rate, False Positive Rate, etc. Reporting sketches quantitative measures into different graphical representation, e.g. ROC curve, F-measure.

- Execution: As depicted in Figure 2, Execution Factory manages the execution process of multiple Docker images of IPAs. It loads the IPAs Docker images stored in AEIPA Repository, runs the operations in the IPA Docker image, stores the visual results (e.g. binary images from detection algorithms, X, Y locations from tracking algorithms, etc.) in the Repository for Evaluation.

- Evaluation: Since the output of the Execution submodule is only visual results, quantitative results are required for an accurate comparison of IPAs. The Evaluation submodule allows the visual results of IPAs to be examined according to a set of standard evaluation matrices. Typically, the Evaluation submodule automatically generates evaluation matrices [14]: true positive, false positive, true negative, false negative, true positive rate, false positive rate, precision and recall, for the visual results of IPAs.

- Repoting: The reporting submodule is concerned with representing the output results graphically. ROC, F-measure, Precision-Recall curves, comparison tables, statistical plots, resources consumption, performance graphs, and pie charts can be displayed according to users choices. Here, the reporting submodule includes a set of scripts to pass the values of the evaluation matrices to Matlab for drawing, and then returning the drawings to the users.

3) Docker Engine: AEIPA system adopts Docker Engine to make: (1) IPAs self-contained, and (2) AEIPA to be cross

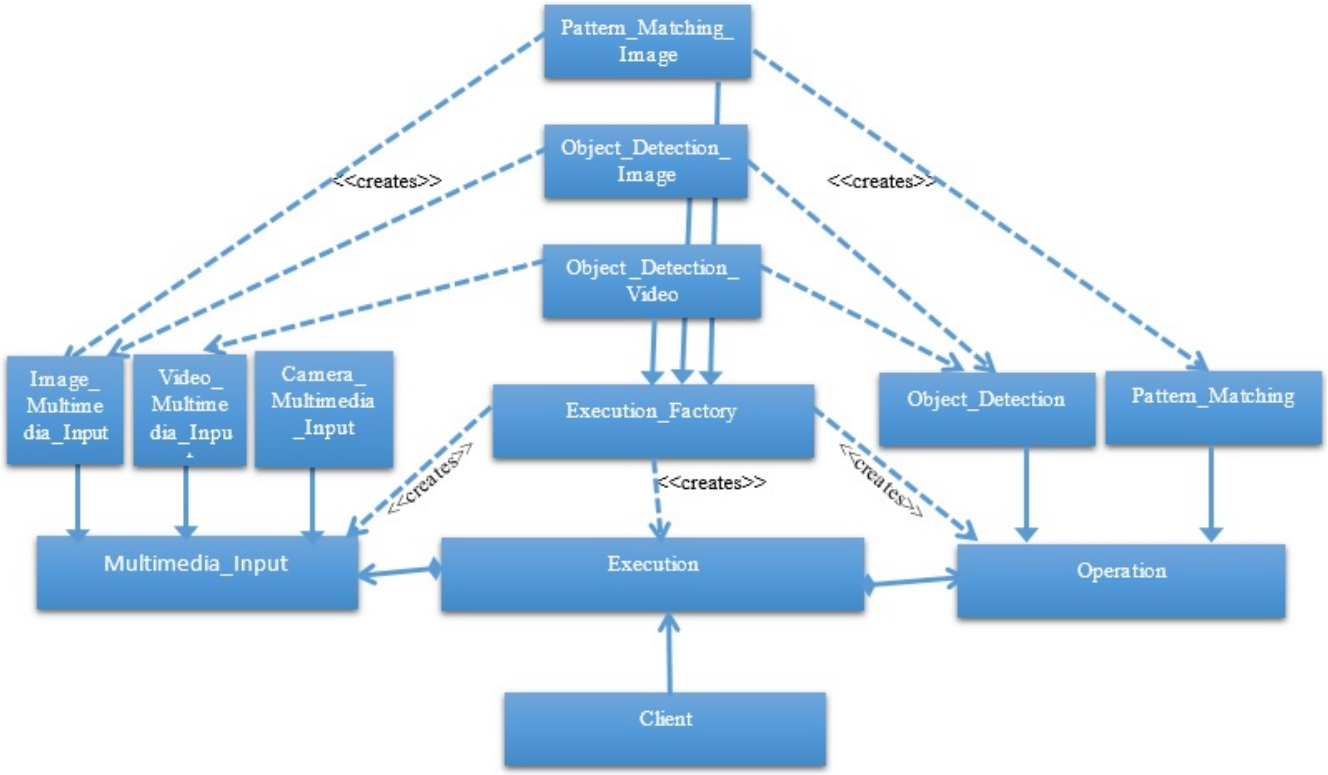

Fig. 2. Execution Design

platform.

AEIPA Supplier adopts Docker Engine to convert IPAs source code into a Docker image ready for execution without need for customization. To allow more agility and mobility of the AEIPA, it is converted into a Docker image. Hence, AEIPA can be used on different operating systems. Also, users are kept away from any installation or environment configuration problems.

IV. CONCLUSION AND FUTURE WORK

AEIPA allows users to execute IPAs with little effort. It manages automatic comparison of related IPAs without customizing IPAs according to the working environment. It adopts Docker containers to wrap AEIPA with its dependencies, and the IPAs into Docker images. Another main advantage of the system is its extensibility. The proposed system can accommodate evaluation any algorithm. For now, the system is being tested using image processing algorithms. The first iteration of the system (system design) has been proposed in this paper.

The future work of this paper is to distribute the execution over multiple servers with no user intervention. Each one of the IPAs being executed may run on a separate server in parallel with the other IPAs, and that is one of the advantages of using Docker. This will speed up the execution and evaluation processes.

REFERENCES

[1] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," IEEE transactions on Intelligent transportation systems, vol. 1, pp. 108-118, 2000.

[2] T. Brosnan and D.-W. Sun, "Improving quality inspection of food products by computer visiona review," Journal of food engineering, vol. 61, pp. 3-16, 2004.

[3] M. Trajkovic, A. J. Colmenarez, S. Gutta, and K. I. Trovato, "Computer vision based parking assistant," ed: Google Patents, 2004.

[4] W. W. Boles and B. Boashash, "A human identification technique using images of the iris and wavelet transform," IEEE transactions on signal processing, vol. 46, pp. 1185-1188, 1998.

[5] CVonline: Image Databases. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm

[6] R. K. Barik, R. K. Lenka, K. R. Rao, and D. Ghose, "Performance analysis of virtual machines and containers in cloud computing," in Computing, Communication and Automation (ICCCA), 2016 International Conference on, 2016, pp. 1204-1210.

[7] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," in Performance Analysis of Systems and Software (ISPASS), 2015 IEEE International Symposium on, 2015, pp. 171-172.

[8] M. G. Xavier, M. V. Neves, F. D. Rossi, T. C. Ferreto, T. Lange, and C. A. De Rose, "Performance evaluation of container-based virtualization for high performance computing environments," in Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on, 2013, pp. 233-240.

[9] A. M. Joy, "Performance comparison between Linux containers and virtual machines," in Computer Engineering and Applications (ICACEA), 2015 International Conference on Advances in, 2015, pp. 342-346.

[10] W. Li and A. Kanso, "Comparing containers versus virtual machines for achieving high availability," in Cloud Engineering (IC2E), 2015 IEEE International Conference on, 2015, pp. 353-358.

[11] T. Kmrinen, Y. Shan, M. Siekkinen, and A. Yl-Jski, "Virtual machines vs. containers in cloud gaming systems," in Proceedings of the 2015

International Workshop on Network and Systems Support for Games, 2015, p. 1.
[12] R. Dua, A. R. Raja, and D. Kakadia, "Virtualization vs containerization to support paas," in Cloud Engineering (IC2E), 2014 IEEE International Conference on, 2014, pp. 610-614.
[13] T. Bui, "Analysis of docker security," arXiv preprint arXiv:1501.02967, 2015.
[14] D. Powers, Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation, Journal of Machine Learning Technologies, vol. 2, no. 1, pp. 3763, 2011.