Software Engineering and Distributed Computing in image processing intelligent systems: a systematic literature review

Luis Jacome-Galarza, Monica Villavicencio-Cabezas, Miguel Realpe-Robalino, Jose Benavides-Maldonado

***Abstract*—Deep learning is experiencing an upward technology trend that is revolutionizing intelligent systems in several domains, such as image and speech recognition, machine translation, social network filtering, and the like. By reviewing a total of 80 studies reported from 2016 to 2020, the present article evaluates the application of software engineering to the field of intelligent image processing systems, it also offers insights about aspects related to distributed computing for this type of systems. Results indicate that several topics of software engineering are mostly applied when academics are involved in developing projects associated to this kind of intelligent systems. The findings provide evidences that Apache Spark is the most utilized distributed computing framework for image processing. In addition, Tensorflow is a popular framework used to build convolutional neural networks, which are the prevailing deep learning algorithms used in intelligent image processing systems. Also, among big cloud providers, Amazon Web Services is the preferred computing platform across the industry sectors, followed by Google cloud.**

***Index Terms*—Image processing, software engineering, deep learning, intelligent vision systems, cloud computing.**

I. Introduction

Image processing is the application of a set of techniques and algorithms to a digital image, in order to analyze, enhance, or optimize its characteristics such as brightness and contrast. Most image processing techniques involve treating the image as either a signal or a matrix and applying standard signal-processing or matrix manipulation techniques [1]. Image processing has numerous applications in most human activities, like medicine, security, astronomy and quality control [2-5]. Image processing applications may benefit from deep learning algorithms, and multithreading.

Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These models have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics [6]. However, the benefits of deep learning need to be assessed with respect to its computational cost [7] and quality. A plethora of studies have shown that state-of-the-art deep learning systems suffer from defects and vulnerabilities that can lead to severe loss and tragedies, especially when applied to real-world safety-critical applications [8]. Therefore, the development of artificial intelligence applications in real world environments is not trivial and the development process plays an important role. This situation has led to a growing interest and need to understand how AI-enabled applications are developed, deployed, and maintained in real-world commercial settings [9]. Deep learning software is mostly developed and trained on the cloud or PCs with powerful GPU support [8]. When it is deployed on a mobile or edge computing device with limited computation power, the deep learning software must be optimized for computation/energy efficiency, which could introduce defects or lead to behavior differences [8].

Another aspect to consider when developing AI applications is multithreading, which refers to the partitioning of the functionality of the application into simple tasks -or threads- to execute them concurrently [10]. The idea of multithreading image processing applications on multicore systems and its gains in performance have been demonstrated by a number of authors who are actively working in distributed architectures [11].

Global IT companies have been investing in distributed architectures for supporting artificial intelligence technology under cloud-based corporate strategy. Amazon's Amazon Web Service (AWS), MicroSoft's Azure Platform, Google's GCP (Google Cloud Platform) and IBM's Bluemix Service build Big Data, IoT, machine learning, deep learning, and cognitive service on top of cloud services [12].

The efficiency of intelligent systems also relies on the use of specific microprocessors in the cloud. The learning phase of deep neural networks relies on GPUs (Graphic Processing Units) processors that were initially designed for video games, such as those by NVIDIA. Other large AI companies often develop dedicated processors, such as Google's TPU (Tensor Processing Unit) or Microsoft's FPGA (Field Programmable Gate Array) [12].

For building intelligent systems, there are also open source deep learning frameworks such as Caffe, Chainer, Deeplearning4J, Keras, CNTK, MatConvNet, Minerva, Mxnet, Purine, Tensorflow, Theano, Torch [12].

State-of-the-art

The state of the art for computer vision tasks is changing quickly, the use of deep learning has given significant advances in those tasks. Among the deep learning models with best performance are the following: for semantic segmentation that is clustering parts of an image that correspond to the same object, we have HRNet-OCR [13], Efficient-Net-L2+NAS-FPN [14], ResNeSt-269 [15], VMVF [16]. For image classification, which is a task that assigns a label for each image, we find FixEfficientNet [17], BiT-L [18], Wide-ResNet-101 [19],

Branching/Merging CNN+ Homogeneous Filter Capsules [20]. For object detection that is a task in which the model finds instances of a class within an image, we have Efficient-Det D7x [21], RODEO [22], Patch Refinement [23], IterDet [24]. For image generation that are the models which take a class as an input and try to obtain its features composing an image, we have NCNS [25], Image Transformer [26].

For its part, in the software engineering component, the systematic literature review in [27] reports that reasoning under uncertainty, search-based solutions and machine learning are the intelligent approaches that mostly implement the agile software development. The work is [27] also reveals that effort estimation, requirements prioritization, resource allocation, requirements selection for a release or sprint and requirement management are the most considered goals of software engineering in intelligent systems.

Research goal

This research seeks to find out how software engineering is applied in projects that involve image processing intelligent systems (i.e. intelligent vision systems) as well as to get insights about aspects related to distributed computing to perform massive-scale and complex computing for this kind of systems. On the one hand, software engineering is important to ensure the quality of intelligent systems by following a systematic approach. Since the automation of processes is being implemented in a wide variety if industries and business, it is crucial to follow a systematic approach when building and deploying this kind of systems. It is important to remark that deep learning projects involve additional challenges like complexity, high computing demand, expensive and time consuming training processes, etc. On the other hand, distributed computing is relevant for this research because deep learning networks are typically computationally expensive to train, requiring long periods of computation on many GPUs; as a result, many users outsource the training procedure to the cloud or rely on pre-trained models that are then fine-tuned for a specific task [28] [29].

The rest of this paper is structured as follows: section 2 describes the research methodology; section 3 summarizes the results of this ongoing research work; section 4 presents the limitations and future work, and section 5 concludes the article.

II. Methodology

For performing this preliminary study, we used the methodology proposed by Kitchenham [30], which includes: the formulation of research questions; the search process; inclusion and exclusion criteria; data extraction; and data analysis and classification.

A. Research questions

The purpose of this research work is twofold: 1) Identify how software engineering (SE) is applied to intelligent vision systems (i.e. systems that involve computer vision and deep learning); and 2) Identify distributed architectures used for this kind of systems. Hence, the research questions are:

RQ1: What software engineering topics are being used in the development of image processing systems with deep learning?

RQ2: What distributed architectures are being used for building image processing systems with deep learning?

B. Research process

In order to answer the RQ1 and the RQ2, we conducted a manual search on the ScienceDirect, Google Scholar and Springer databases. The search was perform in September 2020 and was run on titles, keywords and abstracts, but if the information was not found in those sections, we read the results and conclusions.

The search string for RQ1 (SE areas in image processing with deep learning) and RQ2 (cloud computing architectures for image processing with deep learning) are the following:

RQ1: ”image processing” AND ”deep learning” AND ”software engineering”.

RQ2: ”image processing” AND ”deep learning” AND (”cloud provider” OR ”cloud computing”)

C. Inclusion and Exclusion Criteria

We consider only articles published between 2016 and 2020. The language for the search process was restricted to English, and the type of publication to Scientific Articles.

The inclusion criterion considered in both search processes is: The article should contain topics such as cloud computing, computer vision, deep learning, any software engineering area, and preferably be related to an industry sector.

D. Data Extraction

From the first search process (SE topics in image processing with deep learning), we obtained the following amount of articles:

- Springer: 1899
- Google Scholar: 6810
- ScienceDirect: 450

And from the second search process (cloud computing architectures in image processing with deep learning), the following amount of articles:

- Springer: 961
- Google Scholar: 7340
- ScienceDirect: 481

E. Data Analysis and Classification

The data analysis and classification steps were conducted based on the previously defined inclusion criterion and the classification process explained below until we achieved 40 articles per each research question:

- Reading the abstracts and conclusions of each retrieved article.
- Searching for each criterion within the complete content of the articles.
- When necessary, reading the whole article.
- Classifying by search criteria until 40 articles were achieved per each research question.

The second and third steps were decisive to verify all criteria and avoid discarding articles.

Therefore, the extraction process resulted in a total of 80 articles, which are included in the Appendix.

III. Results

After reviewing the articles, we came up with the following results: we analyzed a total of 80 articles, the distribution of the publishing years is shown in Fig. 1:

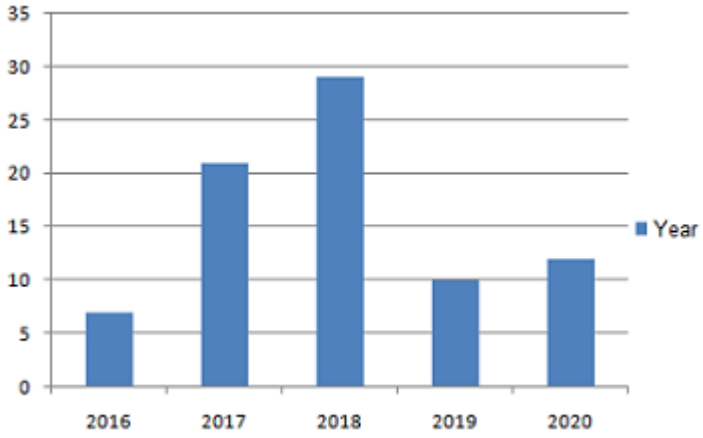

Fig. 1. Distribution of the publishing years of the analyzed articles.

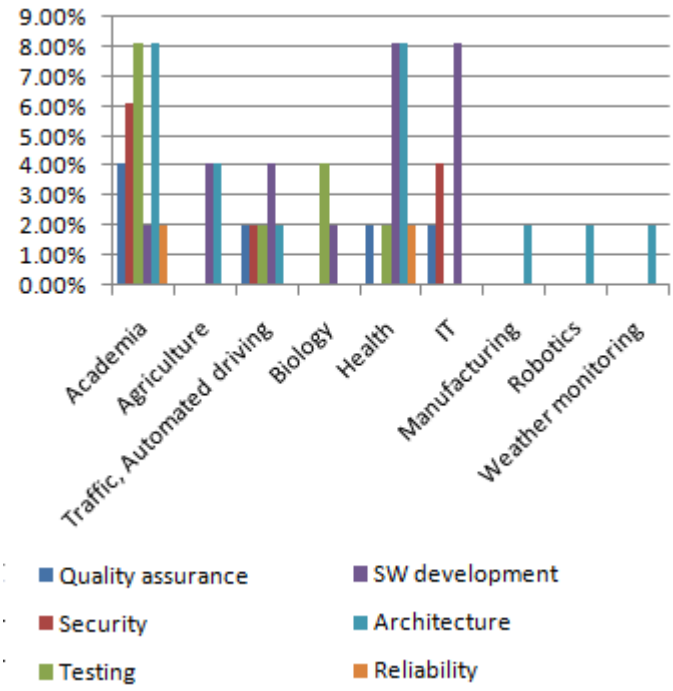

Fig. 2. Software engineering topics involved in deep learning and image processing projects per industry.

A. Query results for RQ1 - SE topics in image processing with deep learning

The results for the first research question are presented in Fig. 2 where it can be observed the software engineering topics used in image processing projects that apply deep learning algorithms classified by industry type.

The topics more commonly referred are software architecture and software development process models. This could be due to the importance of system architecture in AI application development and the increasing use of agile development methodologies.

In the figure, it can also be seen that most of the SE topics have been reported in university projects, which is understandable given the awareness of academics about the importance of applying a systematic approach when developing complex systems. In the case of software testing, this topic also shows its relevance in the automated driving, healthcare and biology industries because testing helps to ensure quality in vision systems.

In the following paragraphs, we provide a series of examples of the use of software engineering in computer vision systems for some industrial sectors.

Health

In S1-034, a deep learning model for early detection of breast cancer using mammographic imaging is presented. The algorithm was formulated applying SE to guarantee scalability, flexibility and reliability. The SE process model used consists of the following phases: collecting requirements for classifying the images into infected and non-infected; designing the initial version of the proposed algorithm and obtaining feedback between phases one and two to confirm that the initial design is in accordance with the required constraints; coding the algorithm; testing the proposed algorithm on data sets corresponding to different case studies. Regarding the testing stage, it was carried out in two parts; the first tested the accuracy of the deep learning algorithm, and the second tested the entire system.

Agriculture

In document S1-030, it is explained that plant diseases are a great threat in agricultural production due to the losses they cause. The severity of plant diseases is an important parameter to measure the level of disease and can therefore be used to predict its progress and recommend treatment. A useful approach is transfer learning because a powerful classification network is built using few data, adjusting the parameters of a previously trained network on a large data set (e.g. ImageNet and the PlantVillage dataset). Finally, it is explained that other tools are important for building specialized software for plant contol like the ”Assess: Image Analysis software helpdesk” which is the most commonly used and also the discipline-standard program to estimate disease severity.

Automotive

In the paper S1-009, authors have devised a framework that supports the development of AI applications for autonomous driving that include Deep Neural Networks (DNN). The framework leads to a robust and disciplined development lifecycle by following five steps: 1. Identifying DNN requirements 2. Developing the learning-algorithm 3. Training the DNN 4.

Validating the training of the DNN 5. Validating the DNN

Additionally, the paper recommends the use of the W model for deep learning using DNNs. This model conceptually integrates a V model for data development with the standard V model used for software development. The term "programming by example" is used to remark the importance of data in developing systems based on deep learning technology.

Finally, the article refers to some standards for developing autonomous driving applications. The most relevant and influential standards for deep learning are Automotive SPICE (Software Process Improvement and Capability Determination) and ISO 26262. The first provides a process framework that structures, at a high abstraction level, software development activities; the second targets safety-related automotive development. Other standar is the ISO/AWI PAS 21448 which addresses the fact that for some ADAS (Advanced Driver Assistance Systems) applications, a fault free system can still suffer from safety violations (e.g. a false-positive detection of an obstacle by radar)

Academia

In the paper S1-001, authors define Secure Deep Learning Engineering (SDLE) as an engineering discipline of deep learning software production, through a systematic application of knowledge, methodology, practice on deep learning, software engineering and security. SDLE is largely different from traditional software engineering where the decision logic is mostly programmed by human developers. SDLE adopts a data-driven programming paradigm in which the deep learning developer's major effort is to prepare the data to be trained and to design the neural network architecture, although the major life cycle phases could still be shared: 1. Requirement Analysis 2. Data-Label Pair Collection. 3. DNN Design and Training Program Implementation. 4. Runtime Training. 5. Testing and Verification. 6. Deployment 7. Evolution and Maintenance

The SDLE authors point out that security vulnerabilities can happen in almost every step. For instance, for the training related steps, poisoning attacks can easily happen by manipulating training data. In the testing related steps, evasion attacks can take place by perturbing the testing data slightly (e.g. adversarial examples). In addition, when deploying the deep learning software to different platforms or with different implementation frameworks, there are always opportunities for adversaries to generate attacks. Other important software engineering issues to consider are: how to effectively generate tests, how to measure the data quality used in tests, how to test deep learning robustness and vulnerabilities; and many other related matters.

In the paper S1-029, academics analyze deep learning processing in real-time. In this example, the software engineering related concern is that there have not been enough frameworks for distributed deep learning models to process real-time streaming data. In addition, it is not easy to deploy an operating environment, such as an operating system, a runtime virtual environment, and a programming model, to implement a deep learning model inference based on multiple distributed nodes.

Information Technology

In the paper S1-038, the authors present a deep-intelligent framework for online video processing that aims to achieve the non-functional requirements listed below:

- Performance: two types of performance requirements.
  - Batch processing of large volumes of historical video data (e.g. conducting a video summary, which is not considered critical)
  - Real-time processing (e.g. videos of emergency cases, considered critical)
- Availability. The framework must be reliable to allow hardware failures providing dependable services to users.
- Scalability. The cloud must be able to scale up, with the ability to add video-sensing devices and processing nodes dynamically.
- Modifiability. The framework must allow easy changes of its functionalities so that future modifications are easy to develop.
- Portability and usability. The framework must also provide a unified way to manage video equipment and video data, and also support different Linux-style OSs.

The proposed Lambda Architecture includes the following technology:

- Service-oriented architecture (SOA) for managing the complexities of video data processing and the various technologies involved. This will enable modifiability and achieve portability.
- Publish - subscribe for decoupling event consumers and providers and handling events while monitoring the framework's running status. This will help to achieve the framework's availability such that users can detect malfunction states and initiate appropriate responses.
- MapReduce for analyzing enormous volumes of video data. It can efficiently process distributed large datasets in parallel to meet performance, scalability, and availability requirements.
- Shared Data pattern for improving performance.
- Layered architecture. Framework functionalities might evolve separately, and the processing of big video data can also be separated into different types and might evolve independently. So, the framework uses a layered architecture to separate the different concerns. This can support modifiability, portability, and usability

Robotics In the paper S1-021, the authors propose an architecture that shows the promise of parallelism in FPGAs (Field Programmable Gate Array). Their architecture matches the massive parallelism inherent in neural networks for object detection with the fine-grain parallelism of an FPGA hardware. This dramatically reduce processing time in a Q-learning algorithm using a multilayer perceptron and power consumption of 9.7W. The results show up to a 43-fold speed up by Virtex-7 FPGAs compared to a conventional Intel i5 2.3 GHz CPUs. Also, the power consumption can be further reduced by introducing pipelining in the data path.

B. Query results for RQ2 - cloud computing architectures in image processing with deep learning

In this section, we present three tables to summarize the findings related to RQ2. This information is associated with the technology chosen for distributed computing used to develop image processing intelligent systems. As it is known, lot of images are needed for building accurate deep learning models. There is a number of images datasets available for this purpose such as multiclass, health care, numbers, environment (animals included), people (faces and bodies), satellite (included geography), forensic science and others.

Table 1 shows the data management and processing frameworks that have been reported in the sample of articles that we have reviewed. Apache Spark is the most popular framework for distributed architectures among industry sectors in intelligent vision systems, followed by Hadoop. As it can be also observed, when it comes to healthcare, almost all frameworks have been used for intelligent vision systems, including MAMLS from Microsoft.

TABLE I
DATA MANAGEMENT AND PROCESSING VS. IMAGE DATASETS

Image Datasets	***Hadoop***	***Spark***	***MAMLS***	***Elastic cloud***	***IBM Watson***	***Other***
Multiclass	5%	7.5%			2.5%	
Health care	2.5%	2.5%	2.5%	2.5%		10%
Environment		2.5%				2.5%
Satellite images	2.5%					
Forensic science		2.5%				
Geography		2.5%				
People		2.5%			2.5%	10%
Biology						5%
Animals						5%
Surveillance						2.5%
Other	5%	5%		5%		10%

Table 2 shows the participation of the main players in cloud computing: Google cloud, Amazon Web Services (AWS), Microsoft Azure, and IBM cloud, being AWS the provider most commonly reported for managing image datasets. In contrast, 15.78% of the analyzed projects still use their own infrastructure.

Table 3 shows the deep learning frameworks used for image processing. Tensorflow and Keras appear to be popular, especially in academic research. One of the reasons associated to this fact is because they are free to use. Both can be used together for obtaining good results as it was reported in one of the revised articles. Keras is also preferred for its ease of use when building deep learning models, according to the opinions expressed in the analyzed papers. Regarding MAMLS (Microsoft Azure Machine Learning Studio), which is a scalable machine learning platform, it has been used in health care projects.

With respect to the type of neural networks used for processing image datasets, CNN (Convolutional Neural Networks)

TABLE II
CLOUD COMPUTING PLATFORMS VS. IMAGE DATASETS

Image Datasets	**Google cloud**	**AWS**	**Azure**	**IBM cloud**	**Other**	**Private**
Multiclass		2.63%		2.63%	5.26%	5.26%
Health care	2.63%	5.26%	2.63%		7.89%	
Environment		2.63%	2.63%			
Satellite images						2.63%
Forensic science	2.63%					
Geography						2.63%
People		2.63%		2.63%	7.89%	2.63%
Biology	2.63%	2.63%				
Animals	2.63%	2.63%				
Surveillance						2.63%
Other	5.26%	7.89%			10.52%	

TABLE III
DEEP LEARNING FRAMEWORKS VS. IMAGE DATASETS

Image Datasets	**Keras**	**Tensorflow**	**Caffe**	**MAMLS**	**OpenCV**	**Other**
Multiclass	2.38%	7.14%	4.76%		2.38%	4.76%
Health care				2.38%		11.90%
Environment						4.76%
Satellite images						2.38%
Forensic science	2.38%					
Geography						2.38%
People	2.38%	4.76%			4.76%	7.14%
Biology		2.38%				2.38%
Animals		4.76%				
Surveillance						2.38%
Other			2.38%		2.38%	16.67%

undoubtely appears as the most utilized network architecture in computer vision projects, being reported in 70.57% in the analyzed papers. CNN architectures are also combined with RNN (Recurrent Neural Networks) and LSTM (Long Short-Term Memory) architectures for approaching specific computer vision problems such as violence detection in videos. In addition, DBN (Deep Belief Network) was found in very specific projects of image processing in Geography.

Regarding system architecture, we found that 14.29% of the analyzed projects reported the use of containers and 14.28% used virtual machines. For the cluster orchestration method, 11.44% used kubernetes, 5.71% Mesos and 5.72% Docker.

Next, we present some examples related to cloud computing architecture applied to intelligent image processing systems:

Paper S2-001 mentions that deep learning has remarkable performance, but it is computationally demanding. Therefore, cloud computing is the answer to the challenges surrounding deployment of deep learning in a laboratory setting, since most deep learning models have over a million parameters each of which must be tuned during the training process. Containerization (i.e. the encapsulation of software into a container with its computing environment) allows for software

to be shipped with proper versioning of dependencies.

Paper S2-008 exposes the fact that the "big-data revolution" has struck biology, because nowadays, it is common for robots to prepare cell samples and take thousands of microscopy images. Looking at the resulting images by eye would be extremely tedious, not to mention subjective. It is also computational consuming; hence, it is highly recommended to enable running pre-trained deep learning models in the cloud in order to improve developing time.

Other revised papers also state the benefits of cloud computing, as it has become a prevalent platform to run deep learning applications. With cloud services, deep learning models can be deployed easily for the convenience of users (S2-011); cloud computing can be adopted to make the system faster and allow many concurrent users (S2-015).

Concerning the datasets utilized in the analyzed papers we can mention: Lung Image Database Consortium (LIDC), Image Database Resource Initiative (IDRI), CT Medical Image, DIARETDB datasets (Health), Fei Face, IIITD Latent Fingerprint, Violent Interaction datasets (Security), PETS, KITTI (Traffic), STL-10, CIFAR-100, ImageNet, ResNet-50 datasets (General), MNIST (Others).

It was also found in the analyzed papers that health care datasets are expensive, difficult to obtain and involve privacy issues; wireless sensor datasets are noisy and have missing values, and industry datasets have complex multimodality features; it is necessary to overcome those challenges to build high quality intelligent systems.

IV. Limitations and future work

The limitation of this research is the definition of broad categories of software engineering topics for presenting the results. As future work, we will expand this research in the specific field of Generative Adversarial Networks applied to image processing.

V. Conclusions

Despite its complexity, the use of deep learning in production environments is getting spread among many industries. In this paper, we analyze how software engineering is applied in intelligent computer vision systems, specifically those that used deep learning algorithms. According to our findings, academia, automated driving and health care industry sectors are the ones were software engineering is mostly applied; we can explain this fact because this type of systems need to be built with a disciplined approach assuring and controlling their quality. For data management and processing, we found that Apache Spark is the most utilized framework, followed by Hadoop. Among the big cloud platforms, Amazon Web Services is the most widely used in all industry sectors, followed by Google cloud. Also, using Tensorflow (powerful) with Keras (easy-to-use) is a good combination for building deep learning models from image datasets. Finally, we conclude that it is very important to have good quality image datasets for training reliable deep learning systems that can be used in production environments. Moreover, the systematic approach of software engineering has been useful to define process models focused on the improvement of the development of intelligent computer vision systems.

References

[1] P. Ganney, E. Claridge, in Clinical Engineering, Image Processing Software, 2014.

[2] A. Papour, Z. Taylor, O. Stafsudd, W. Grundfest. "Real time early detection imaging system of failed wounds and heterotopic ossification using unique Raman signatures". In Medical Imaging, Biomedical Applications in Molecular, Structural, and Functional Imaging . International Society for Optics and Photon-ics, March 2015, Vol. 9417, pp. 94171G.

[3] L. Bao, Y. Zhou. "Image encryption: Generating visually meaningful encrypted images". Information Sciences, 324, 2015, pp. 197-207.

[4] M. Perrin, J. Maire, P. Ingraham, D. Savransky, M. Millar-Blanchaer, S. Wolff, C. Marois. "Gemini Planet Imager observational calibrations I: Overview of the GPI data reduction pipeline". In Ground-based and Airborne Instrumentation for Astronomy" V (Vol. 9147, pp. 91473J). International Society for Optics and Photonics, 2014.

[5] H. Sereshti, Z. Poursorkh, G. Aliakbarzadeh, S. Zarre, S. Ataolahi. "An image analysis of TLC patterns for quality control of saffron based on soil salinity effect: A strategy for data (pre)-processing". Food chemistry, 239, 2018, pp. 831-839.

[6] P.S. Martins, J. Real. "Minimizing the mode-change latency in real-time image processing applications". In Bio-Inspired Computation and Applications in Image Processing, 2016.

[7] Image Processing, https://www.sciencedirect.com/topics/engineering/image-processing, last accessed 2020/04/21.

[8] L. Ma, F. Juefei-Xu, M. Xue. Secure Deep Learning Engineering: A Software Quality Assurance Perspective. arXiv preprint arXiv:1810.04538, 2018.

[9] I. Targio, I. Yaqoob, N. Anuar, S. Mokhtar, A. Gani, S. Khan. "The rise of big data on cloud computing: Review and open research issues", 2015, vol. 47, p. 98-115

[10] W. Fu, T. Menzies. "Easy over Hard: A Case Study on Deep Learning", In Proceedings of the 2017 11th joint meeting on foundations of software engineering, 2017, pp. 49-60.2017.

[11] L. Lwakatare, A. Raj, J. Bosch, H. Holmstrom, I. Crnkovic. "A Taxonomy of Software Engineering Challenges for Machine Learning Systems: An Empirical Investigation", 2019.

[12] K. Lee, N. Ha. AI platform to accelerate API economy and ecosystem. 2018 International Conference on Information Networking (ICOIN). IEEE, 2018. p. 848-852.

[13] Tao, A., Sapra, K., Catanzaro, B. (2020). Hierarchical multi-scale attention for semantic segmentation. arXiv preprint arXiv:2005.10821.

[14] Xie, Q., Luong, M. T., Hovy, E., Le, Q. V. (2020). Self-training with noisy student improves imagenet classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10687-10698).

[15] Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Zhang, Z., Lin, H., Smola, A. (2020). Resnest: Split-attention networks. arXiv preprint arXiv:2004.08955.

[16] Mandelli, S., Borra, F., Lipari, V., Bestagini, P., Sarti, A., Tubaro, S. (2018). Seismic data interpolation through convolutional autoencoder. In SEG Technical Program Expanded Abstracts 2018 (pp. 4101-4105). Society of Exploration Geophysicists.

[17] Touvron, H., Vedaldi, A., Douze, M., Jgou, H. (2020). Fixing the train-test resolution discrepancy: Fixefficientnet. arXiv preprint arXiv:2003.08237.

[18] Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., Houlsby, N. (2019). Big transfer (bit): General visual representation learning. arXiv preprint arXiv:1912.11370, 6(2), 8.

[19] Chen, L. C., Wang, H., Qiao, S. (2020). Scaling Wide Residual Networks for Panoptic Segmentation. arXiv preprint arXiv:2011.11675.

[20] Kalganova, T., Byerly, A., Dear, I. (2020). A Branching and Merging Convolutional Network with Homogeneous Filter Capsules.

[21] Tan, M., Pang, R., Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10781-10790).

[22] Acharya, M., Hayes, T. L., Kanan, C. (2020). Rodeo: Replay for online object detection. arXiv preprint arXiv:2008.06439.

[23] Lehner, J., Mitterecker, A., Adler, T., Hofmarcher, M., Nessler, B., Hochreiter, S. (2019). Patch Refinement–Localized 3D Object Detection. arXiv preprint arXiv:1910.04093.
[24] Rukhovich, D., Sofiiuk, K., Galeev, D., Barinova, O., Konushin, A. (2020). IterDet: Iterative Scheme for ObjectDetection in Crowded Environments. arXiv preprint arXiv:2005.05708.
[25] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., Poole, B. (2020). Score-Based Generative Modeling through Stochastic Differential Equations. arXiv preprint arXiv:2011.13456.
[26] Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D. (2018, July). Image transformer. In International Conference on Machine Learning (pp. 4055-4064). PMLR.
[27] Perkusich, M., e Silva, L. C., Costa, A., Ramos, F., Saraiva, R., Freire, A., Perkusich, A. (2020). Intelligent software engineering in the context of agile software development: A systematic literature review. Information and Software Technology, 119, 106241.
[28] T. Gu, B. Dolan-Gavitt, S. Garg. Badnets: "Identifying vulnerabilities in the machine learning model supply chain". arXiv preprint arXiv:1708.06733, 2017.
[29] A. Parvat, J. Chavan, S. Kadam. "A survey of deep-learning frameworks". In International Conference on Inventive Systems and Control (ICISC). IEEE, 2017, pp. 1-7.
[30] B. Kitchenham, S. Charters. "Guidelines for performing systematic literature reviews in software engineering". Cs. auckland. ac. nz., 2016.
[31] P. Moritz, et al. "Ray: A Distributed Framework for Emerging AI Applications", 2017.
[32] SparkR https://spark.apache.org/docs/latest/sparkr.html, last accessed 2020/04/20.
[33] N. Madhavji, A. Miranskyy, K. Kontogiannis. Big picture of big data software engineering: with example research challenges. In 2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering, 2015, pp. 11-14.
[34] Q. Xiao, K. Li, D. Zhang, W. Xu. "Security risks in deep learning implementations". In : 2018 IEEE Security and Privacy Workshops (SPW). IEEE, 2018, pp. 123-128.
[35] H. Hosseini, B. Xiao, A. Clark, R. Poovendran. "Attacking automatic video analysis algorithms: A case study of google cloud video intelligence api". In : Proceedings of the 2017 on Multimedia Privacy and Security. ACM, 2017, pp. 21-32.
[36] B. Kehoe, S. Patil, P. Abbeel, K. Goldberg. "A survey of research on cloud robotics and automation". IEEE Transactions on automation science and engineering, vol. 12, no 2, 2015, pp. 398-409.

APPENDICES

The list of the analyzed papers is available in the Appendix available in:

https://www.dropbox.com/s/ifpigan0e3s79ck/appendices.pdf?dl=0