A new features vector matching for big heterogeneous data in intrusion detection context

Marwa Elayni, Ouajdi Korbaa, Farah Jemili, Basel Solaiman

5th International Conference on Advanced Technologies
For Signal and Image Processing, ATSIP' 2020
September 02-05, 2020, Sfax, Tunisia

A new features vector matching for big heterogeneous data in intrusion detection context

Marwa Elayni
Universite de Sousse, ISITCom
MARS Research Laboratory, LR17ES05
Hammam Sousse 4011, Tunisia
aynimarwa@gmail.com

Farah Jemili
Universite de Sousse, ISITCom
MARS Research Laboratory, LR17ES05
Hammam Sousse 4011, Tunisia
jmili_farah@yahoo.fr

Ouajdi Korbaa
Universite de Sousse, ISITCom
MARS Research Laboratory, LR17ES05
Hammam Sousse 4011, Tunisia
ouajdi.korbaa@centraliens-lille.org

Basel Solaiman
Universite de Bretagne, IMT Atlantique, Pays de la loire
Latim INSERM U650
Brest, Bretagne
basel.solaiman@imt-atlantique.fr

***Abstract*— Nowadays, the volume of data considerably increasing, the data is exploding on the scale of the Exabyte and the Zettabyte at an exceptionally high rate. These can be characterized as big data. Hence, the security of the network, Internet, websites, Iot devices and the organizations, of this growth is indispensable. Detecting intrusions in such a big heterogeneous data environment is challenging. In this paper, we will present a new representation of data that can support this big heterogeneous environment. We will use three different datasets and propose an automatically matching algorithm that measures the semantic similarity between each two features existing on different datasets. Thereafter, an approximate vector is created that any type of coming data can be stored. With this representation, we can have subsequently an efficient intrusion detection system that can be able to acknowledge any instance of the existing data in the networks.**

Keywords—Big heterogeneous data, intrusion detection systems, semantic similarity, matching data, Jaccard similarity coefficient

I. INTRODUCTION

Technological evolution in networking, Internet, the Iot devices, the 5G communication media and the worldwide network traffic lead a huge amount of data to be generated every second from heterogeneous sources.

These data are characterized as big heterogeneous data by five criteria: volume, variety, velocity, veracity and visualization [1]. All these data are saved into log files. Each log file contains a specific set of data such as, host log, networks log, wireless networks log and applications log. Therefore, security systems such as Intrusion Detection System are necessary to protect the network computing.

Two methods are used for intrusion detection anomaly detection and misuse detection [2]. The first one is based on dividing the existing behaviors in the dataset ß into two parts {ß and Ь}. In this regard, unsupervised learning is used. Behaviors representing large dataset are classified as normal {ß}. Behaviors representing a small set of dataset are those that seem outliers {Ь}. The misuse detection is based on specifically known behavior observations, called signatures. In this case, the dataset used, must be a labeled dataset {ß}. Supervised learning is used to obtain a model {M} able to predict and detect attempts that are similar to those already learned.

Since the evolution of big traffic networks computer data, it has become difficult to detect intrusion in big data environment [3]. The zero day attacks are a significant topic that represents new attacks when Intrusion Detection System (IDS) doesn't have any information about the new data. It can break through the network system and cause a devastating damage [4]. Moreover, detect intrusion from heterogeneous data which refers to different structures of data coming from many IT infrastructures such as workstations, servers, routers, sensors, Iot devices, smart cities, web servers… is challenging [5] [6]. So, IDS remains as an interesting topic of study because of the increasing speed of networks, the appearance of news attacks, and the rapid change of structure data.

To analyze the available data, many Machine Learning (ML) methods are used for IDS, such as Neural Networks, Decision Tree, Random Forest, genetic algorithm, hidden markov models, Bayesian Networks, Support Vector Machines, and Fuzzy Logic [7] [8] [9] [10] [11] [12]. These methods represent shallow architectures. They are unable to handle just one source of data of intrusion detection. It is difficult with these traditional methods to identify unknown attacks, cannot provide solutions in real time, and cannot deal with the huge amount of data in large datasets. However, big data and distributed technologies including Hadoop, Spark, Pig, Kafka, Flink, NoSql have many benefits such as storing, analyzing and processing big data.

In this work, we will present a new big heterogeneous data modeling for intrusion detection that can contain data of various types.
The idea is to present a features vector that represents a standard of a finished set of the existing features in IT infrastructures. Dealing with this representation enables to learn different types of data coming from different sources. Our system will be efficient, autonomous and able to learn automatically any input of data then store it in a train and a test dataset.

To our best knowledge, the idea is to represent a standard of the existing features that is not discussed in the literature and we will be the first to propose this idea to deal with the big heterogeneous traffic of intrusion detection.

This suggested approach is applied on three types of data that are collected from different devices, namely CICIDS 2017, UNSW-NB 15 and NSL-KDD.

This paper is organized as follows: section 2 presents the related works. Section 3 presents a general formalism of our approach. Section 4 gives details of our contribution. Section

978-1-7281-7513-3/20/$31.00 ©2020 IEEE

5 demonstrates and discusses the obtained results. Finally, Section 6 concludes the paper and presents some recommendations for future work.

II. RELATED WORKS

Recent years have seen a trend of proposing a various approaches and distributed architectures to handle dynamic intrusions in a big data environment.

Faker et al. [13] present a multi-classifier detection system that contains, Deep Feed-Forward Neural Network (DNN), Random forest and Gradient Boosting Tree (GBT). In the evaluation step, they use two datasets UNSW NB15 and CICIDS 17 then for each dataset they apply their detection system. The results show a high accuracy with DNN on UNSW NB15, and with BGT on CICIDS 17.

Hassan et al. [14] propose a hybrid deep learning detection system based on a CNN and a WDLSTM network to retain long-term dependencies among extracted features doing with a CNN to prevent over fitting on recurrent connections. For the evaluation, UNSW-NB 15 dataset is used.

Elayni et al. [15] present a survey that review seven works deploy interesting approaches in the context of big data environment. These works aim to propose a detection system that focuses on pre-processing step, the feature selection and the detection method with big data technologies such as parallel programming MapReduce, Spark with Machine learning libraries Mlib, Hadoop with Mahoot … the datasets used in these reviewing papers are kdd99 and Mawilab and UNSW-NB 15.

Resende et al. [16] review 35 works dated 2005 to 2017 for intrusion detection that used the Random Forest for the detection process. Among these works, we find that each approach is applied only on one type of data, such as ad hoc traffic (9 papers) and kdd99 (26 papers).

All these studies have considered different approaches for improving data quality or for learning data. They considered a distributed architecture in a big data context using several big data tools, (big data storage tools, big data processing tools, big data streams tools). The result of these studies has shown effective results in terms of detection rate, speed, real time.

On the other hand, these works are limited. This limitation is the actual IDS, until today, it is able to react only with the observations that it had learned during the classification phase. If, we have a new observation with a different structure, the system will not take it into consideration.

III. APPROACH FORMALISM

Our work aims to propose a new representation of a universal vector which will identify all the possible features that may exist in IT infrastructures. To justify the use of this new representation, we should answer the two following questions;

why we propose this new representation which differs from the traditional method that it is only interested in a single type of data which comes from a well-defined source? And how, we will present our model?

To respond to these questions, we should identify our needs.

Traditional IDS handle a single type of data that has the same structure. However, as it is described in the previous section, even the studies that focus on big data use the same data used in ancient works.

The data come from different IT infrastructures such as, workstations, routers, sensors, servers, Iot devices and etc. At each level of component, a set number of features are well known and with these features, we can identify an observation or a vector connection.

All observations that have the same structure are grouped on the same category. In the network context, generally each level has a well-defined number {n} of features {f}, also, a number of common features that identify any connection as source IP address, destination IP address, destination port, source port, protocol, duration, … are still present in different categories.

So, our idea is to define a new universal vector U = {u_1, u_2… u_n} which include all the categories observable on an any IT component whatever its type.

The figure below explains clearly the idea which includes many types of devices, each device generates a traffic that is stored in log file then in a database. At each level of devices, each dataset ß$_i$ has a number k of features, a features vector is identified {X, Y, Z… W}, and each one has a different structure from the others.

The universal vector will be generated and will have m features with

$$\begin{cases} m <= (n_1+n_2+n_3) \\ f_m \in \text{or} \notin \text{ to the set of X features} \\ f_m \in \text{or} \notin \text{ to the set of Y features} \\ f_m \in \text{or} \notin \text{ to the set of Z features} \\ \dots \\ f_m \in \text{or} \notin \text{ to the set of W features} \end{cases}$$

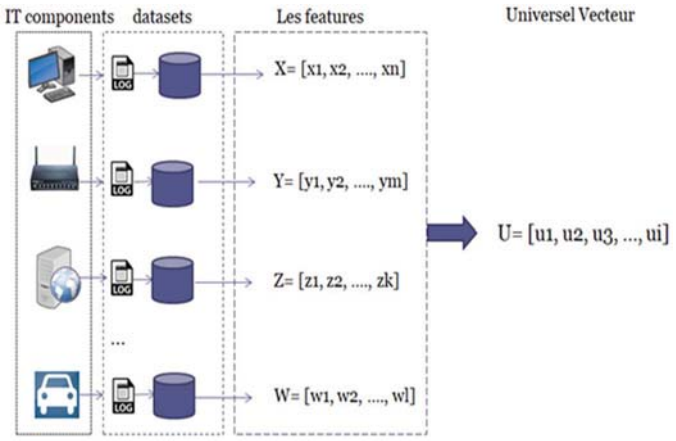

Fig. 1. Approach formalism with an example of different IT infrastructures

IV. CONTRIBUTION

In this part, we describe our approach, in our case; we take three different types of data {X, Y, and Z}. Then, we present for these data an approximate features vector which is going to be extensible and in the case, if we have a new features vector {W} we can update it. We start by describing the features datasets, then, we describe the measure of similarity used in our approach. Finally, we present our matching algorithm for generating the approximate vector.

A. Features dataset description

- **CICIDS 17 dataset [17]:** it contains benign and the most up to date common attacks, this benchmark presents realistic background traffic. It present the traffic of 25 users based on the HTTPS, HTTP, SSH, FTP, email protocols. A number of attacks are implemented for simulation such as, brute force FTP, DOS, Heartbleed, Botnet, Brute force SSH. For this dataset, 78 features are extracted using CICFlowMeter..
- **NSL-KDD dataset [18]:** to solve the problem of the KDD dataset which doesn't represent the existing real networks traffics. Many improvements are suggested in the new NSL-KDD dataset. The mainly ones focus on not including redundant records and on the elimination of duplicate records in the test sets. In this dataset, we find four categories of attacks, DOS, Probing, U2R, and R2L. It represents traffic of nine weeks of raw TCP Dump data for a local-area Network LAN. A connection is a sequence of TCP packets. There are 41 features which are classified into three categories, the first one represents the basic features such as IP address, protocol, port; the second is for the content features and the latter represents the traffic features.
- **UNSW-NB 15 dataset [19]:** the benchmark dataset UNSW-NB is the most used datasets in recent years. It represents a traffic which is collected in January and February 2015. The IXIA PerfectStorm is used to implement different attacks against several servers. It contains nine categories of attacks types such as DDOS. The features are generated using Bro-IDS and Argus; they are classified into five categories: content features, basic features, time features, flow features and additional originated features.

B. Semantic similarity

Semantic similarity among texts or words is popularly used in the application of Natural language [20]. This measure represents the degree of semantic equivalence between two sentences or documents. It is widely used for text classification, text clustering, machine translation, information retrieval. Many methods exist, including: Jaccard similarity coefficient, cosine similarity, Dice …

In our model, we use the Jaccard similarity coefficient [21] for calculating similarity between each two features to find similar word. It represents the result of division between the number of features that are common to all divided by the number of properties as shown below. Many semantic similarity measures are used in literature for comparing two or more documents. In our case, we need this measure to only calculate the similarity between two simple words. So, we believe that any existing measure will lead to the same result.

$$\text{Jaccard}(X, Y) = |X \cap Y| / |X \cup Y| \quad (1)$$

Jaccard distance is non-similar measurement among data sets. It can be determined by the inverse of the Jaccard coefficient which is obtained by removing the Jaccard similarity from (1).

C. Features vector matching

Consider a features vector $U= \{u_1, u_2 \dots u_n\}$ with three categories cat$\in$ {X, Y, Z}, such that $X= \{X_m\}_{m=1}^{M}$, $Y= \{Y_k\}_{k=1}^{K}$ and $Z= \{Z_L\}_{l=1}^{L}$ are a set of features for each vector. Among these features, we can find common features F_{com} which belong to the vectors X, Y, Z, $F_{com} = \{X \cap Y \cap Z\}$.

For extracting these common features from different vectors, we use a measure similarity between words. The basic idea is to calculate the similarity of each feature relative to others features then each couple that has a high probability will be considered as the same feature.

Algorithm features vector matching

Require: X, Y, Z previously generated from three datasets
Ensure: generate the approximate features vector
{*Initialization*}
$X_m \leftarrow \{\}$
$m \leftarrow 0$
{*Estimate dist of the Jaccard distance between Fi and Fj*}
$U_{ni} \leftarrow \{u_1, u_2 \dots u_n\}$
$Y_{kj} \leftarrow \{y_1, y_2 \dots y_k\}$
$X_m \leftarrow U_{ni}$
$m \leftarrow n$
For i =1 … m
  For j=1 … k
    Dist_jaccard $(U_i, Y_j) \leftarrow 1 - |U_i \cap Y_j| / |U_i \cup Y_j|$
  End for
End for
{*Create the approximate vector*}
  For i =1 … m
    For j=1 … k
  if dist_jaccard $(U_i, Y_j) >= \delta$
  $X_i \leftarrow$ "U_i and Y_j represent the same feature"
  $m \leftarrow n$
  Else
  $m \leftarrow n+1$
  $X_{m+1} \leftarrow Y_i$
    End for
  End for

Based on that, we can define the approximate feature vector $X= \{X_m\}_{m=1}^{M}$ such that its Mth feature contains the label of connection.

The algorithm above explains our contribution, we started with an initialization step that takes as input the features vectors of one of the three datasets {X}, we supposed in this step that the approximate vector {U} takes the same dimensional of the input vector {X}. Then, we estimated the degree of similarity by using the Jaccard distance between the approximate features vector {X} and the new input features vector whatever, in our case we have two news vectors {U, Y}.

The final step aims to create and match the structure of the proximate vector according to previous results of the Jaccard distance. Then, with this new extensible structure we could store all types of data. For each feature that does not

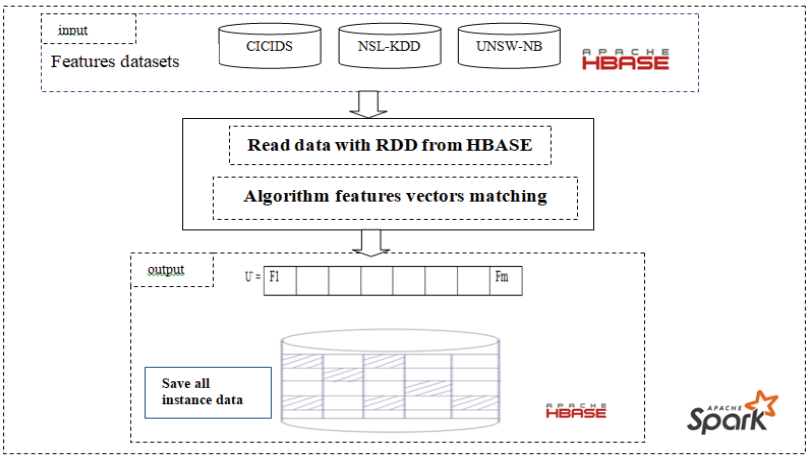

Fig. 2. Approach architecture

TABLE I. JACCARD SIMILARITY MATRIX OF 10 FEATURES WITH EACH OTHER AND FINDING OUT THE MOST SIMILAR FEATURES

	Source IP address	*Source port number*	*Destination IP address*	*Destination port number*	*Protocol*	*Dload*	*Duration*	*service*	*Sload*	*Dloss*
Duration	35,71	33,33	50	46,66	27,27	30	**100**	16,66	30	9,09
Protocol	21,42	38,46	18,75	25	**100**	22,22	27,27	19 ,99	22,22	25
Service	30,76	19,99	26,66	25	19,99	0	16,66	**100**	9,99	0
Flag	7,14	0	6,25	58,82	11,11	28,57	9,09	0	28,57	14,28
Src IP adresse	**81,81**	23,52	46 ,66	21,05	15,38	16,66	21,42	36,36	27,27	8,33
Dst IP adresse	53,84	16,66	**76,92**	41,17	14,28	25	28,57	23,07	15,38	16,66
Land	15,38	6,66	21,42	12,5	11,11	50	33,33	0	0,5	14,28
wrong_fragment	23,52	37,5	35,29	41,17	23,07	15,38	38,46	14,28	15,38	7,6
Urgent	13,33	28,57	26,66	25	19,99	0	27,27	19,9	0	0
Hot	7,69	15,38	14,28	13,33	28,57	14,28	22,22	0	14,28	16,66

exist in a set of data and exist in other set of data, it was saved as missing data. With this representation, it will be relevant to detect intrusion from big heterogeneous data and given thereafter an efficient intrusion detection system that can be able to knowledge any instances of existing data in the networks.

For implementation, we used Spark [22] to process data, Hbase [23] to store data and python such as PySpark for implementing our algorithm. In [24], the authors compared Hadoop and Spark by its architectures and the libraries of Machine Learning. In our case, Spark is more useful than Hadoop as it has an access on NoSQL databases, such as Hbase to store data. Spark will also be very useful in our future works, such as in the detection step by using its Machine Learning libraries MLib.

Figure 2 illustrates the step of our approach in the context of big data environment. At first, each features vector is stored on Hbase on Spark then we create an RDD (Resilient Distributed Datasets). The estimation of the Jaccard distance is greedy at computing resource level, so, by using RDD the processing will be faster. The RDD is the primary underlying data structures of Spark, it's highly fault tolerant. The data is written into multiple executable nodes. RDD does not need any hard disk or any other secondary storage, all that it needs is ram memory.

Subsequently, the proposed algorithm features vector matching is applied for generating the new approximate vector. Finally, all instance data coming from three datasets are saved in Hbase.

V. RESULTS AND EVALUATION

To assess the performance of the proposed method, we conducted our experiment with three types of data, CICIDS, NSL-KDD, UNSW-NB; the first has as features 78, the second 41 and the latter 48. For our experiment, we have used a DELL computer machine with Intel Core (TM) i7- 8550U CPU and 8GO RAM.

TABLE II. NUMBER OF FEATURES AT EACH LEVEL IN OUR APPROACH

Number of CICIDS features	78
Number of UNSW-NB features	48
Number of NSL-KDD features	41
Common features	13
Count_features of approximate vector	154

The table I illustrates the procedure of Jaccard distance described above. In our case, the number of features is so big. So, we will illustrate a matrix result of only the 10 first features. Also, we can notice that a threshold δ is needed for taking decision about common features and for matching with other features of the approximate vector.

When we applied the Jaccard distance, we notice that the most features that are similar have a minimum percentage of the threshold about 70 %. In the first step, we use the NSL-KDD and the UNSW-NB. The column in Table I presents distance similarity of NSL-KDD and line presents distance similarity of UNSW-NB. We can notice in the Matrix that all common features have a high rate, which varied between 70% and 100%. Firstly, the approximate vector X has a global of 83 features including 6 common features. It represents the features of the two first vectors namely NSL-KDD and UNSW-NB. Then, the algorithm compared and matched the new obtained features vector X with the features vector that represents CICIDS features.

In the end, the new approximate features vector after the matching with the later vector is stored on Hbase with 154 features including 13 common features (See Table II).

VI. CONCLUSION AND FUTURE WORKS

In this paper we proposed a new representation of features that represent a big heterogeneous data of intrusion detection.

Detect intrusion from different big heterogeneous sources is challenging, with this new representation this challenge will be possible and relevant. An approximate vector is created automatically whatever the structure and the number of incoming vectors which each vector defines specific networks traffics.

For calculating the number of features of approximate vector, we proposed an algorithm based on measuring the semantic similarity between two features such as Jaccard distance. With the set of 167 features that presented three types of datasets, we have generated a new vector with 154 features that contains a data which come from big heterogeneous sources.

The presented work opens avenues for further investigation. We will investigate at first the detection method by applying different methods of Machine Learning for extracting knowledge from a new big heterogeneous data such as Random forest, bagging, then given an efficient intrusion detection system that can be able to know any instances of the existing data in the networks. In other direction, we will also propose ontology for presenting a well-defined standard which will represent all the existing features at any type of source (router, server, sensor, Iot devices ...).

REFERENCE

[1] B. Bostami and M. Ahmed, "Intrusion Detection for Big Data," Data Analytics, pp. 375–402, 2018.

[2] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," Journal of Network and Computer Applications, vol. 36, no. 1, pp. 25–41, 2013.

[3] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and Big Heterogeneous Data: a Survey," Journal of Big Data, vol. 2, no. 1, 2015.

[4] H. Holm, "Signature Based Intrusion Detection for Zero-Day Attacks: (Not) A Closed Chapter?," 47th Hawaii International Conference on System Sciences, 2014.

[5] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and Big Heterogeneous Data: a Survey," Journal of Big Data, vol. 2, no. 1, 2015.

[6] M. Essid and F. Jemili, "Combining intrusion detection datasets using MapReduce," 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2016.

[7] S. M. Othman, F. M. Ba-Alwi, N. T. Alsohybe, and A. Y. Al-Hashida, "Intrusion detection model using machine learning algorithm on Big Data environment," Journal of Big Data, vol. 5, no. 1, 2018.

[8] K. Park, Y. Song, and Y.-G. Cheong, "Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm," IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService), 2018.

[9] T. Ogino, "Evaluation of Machine Learning Method for Intrusion Detection System on Jubatus," International Journal of Machine Learning and Computing, vol. 5, no. 2, pp. 137–141, 2015.

[10] M. Elayni and F. Jemili, "Using MongoDB Databases for Training and Combining Intrusion Detection Datasets," Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing Studies in Computational Intelligence, pp. 17–29, 2017.

[11] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," 2010 IEEE Symposium on Security and Privacy, 2010.

[12] P. Akshaya, "Intrusion Detection System Using Machine Learning Approach," International Journal Of Engineering And Computer Science, Mar. 2016.

[13] O. Faker and E. Dogdu, "Intrusion Detection Using Big Data and Deep Learning Techniques," Proceedings of the ACM Southeast Conference on ZZZ - ACM SE 19, 2019.

[14] M. M. Hassan, A. Gumaei, A. Alsanad, M. Alrubaian, and G. Fortino, "A hybrid deep learning model for efficient intrusion detection in big data environment," Information Sciences, vol. 513, pp. 386–396, 2020.

[15] M.Elayni, F.Jemili, O.Korbaa and B.Soulaimen, "Big Data processing for Intrusion Detection System context : A review"The International Conference on Intelligent Systems Design and Applications (ISDA) At: Pretoria, South Africa,2019

[16] P.A.A Resende, P and A.C Drummond, "Survey of Random Forest Based Methods for Intrusion Detection Systems", ACM Computing Surveys, vol. 51, 1–36, 2018.

[17] CICIDS 2017: https://www.unb.ca/cic/datasets/ids-2017.html

[18] NSL-KDD : https://github.com/defcom17/NSL_KDD

[19] UNSW-NB 2015: https://github.com/InitRoot/UNSW_NB15

[20] P. Sitikhu, K. Pahi, P. Thapa, and S. Shakya, "A Comparison of Semantic Similarity Methods for Maximum Human Interpretability," Artificial Intelligence for Transforming Business and Society (AITB), 2019.

[21] M. Chahal, "Information Retrieval using Jaccard Similarity Coefficient," International Journal of Computer Trends and Technology, vol. 36, no. 3, pp. 140–143, 2016.

[22] SPARK: http://spark.apache.org/

[23] HBASE: https://hbase.apache.org/

[24] I. Chebbi, W. Boulila, N. Mellouli, M. Lamolle, and I. Farah, "A comparison of big remote sensing data processing with Hadoop MapReduce and Spark," 4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), 2018.