SaFe-NeC: a Scalable and Flexible system for Network data Characterization

Daniele Apiletti, Elena Baralis, Tania Cerquitelli, Paolo Garza, and Luca Venturini
Department of Control and Computer Engineering
Politecnico di Torino, Italy
{name.surname}@polito.it

***Abstract*—Nowadays, large volumes of data and measurements are being continuously generated by computer and telecommunication networks, but such volumes make it difficult to extract meaningful knowledge from them. This paper presents SaFe-NeC, an innovative methodology for analyzing network traffic by exploiting data mining techniques, i.e. clustering and classification algorithms, focusing on self-learning capabilities of state-of-the-art scalable approaches. Self-learning algorithms, coupled with self-assessment indicators and domain-driven semantics enriching data mining results, are able to build a model of the data with minimal user intervention and highlight possibly meaningful interpretations to domain experts. Furthermore, a self-evolving model evaluation phase is included to continuously track the quality degradation of the model itself, whose rebuilding is triggered as soon as quality indicators fall below a threshold of tolerance. The proposed methodology can exploit the computational advantages of distributed computing frameworks, as the current implementation runs on Apache Spark. Preliminary experimental results on a real traffic dataset show the full potential of the proposed methodology to characterize network traffic data.**

I. Introduction

Due to the continuous growth in network speed and service usage, and the increasing number of end users and service providers, even medium-scale networks transfer terabytes of data per day. Thus, two major issues hamper network data analysis: (i) A huge amount of data can be collected even in short time frames. (ii) Traffic and usage patterns rapidly evolve and new behaviors or services appear, making the dataset heterogeneous, unstable, harder to mine and to understand for domain experts. When dealing with Big Data collections, such as the network datasets, the computational cost of the data mining process (and in some cases the feasibility of the process itself) can potentially become a critical bottleneck in data analysis. To date, parallel and distributed approaches have been adopted to increase efficiency and scalability of network traffic mining algoritms [1], [2], [3], [4].

In this paper we argue towards a new methodology of proactive and self-learning network data analytics systems that automatically mine large network datasets with minimal user intervention. The challenge is to devise new efficient and scalable approaches, able to deal with huge network traffic data and to provide meaningful insights.

A significant effort has been devoted to the application of data mining techniques to network analysis. The scopes include studying correlations among data (e.g., association rule extraction for network traffic characterization [5], [6] or for router misconfiguration detection [7]), extracting information for prediction (e.g., multilevel traffic classification [8], Naive Bayes classification [9], analytics and statistical models for LTE Network Performance [10], one-class SVM [11] for intrusion detection), grouping network data with similar properties (e.g., clustering algorithms for intrusion detection [12], for classification [13], [14], for deriving node topological information [15], for identifying classes of traffic [16], for unveiling YouTube CDN changes [17]). While supervised classification algorithms require previous knowledge of the application domain (e.g., a labeled traffic trace), clustering algorithms do not. Hence, the latter is a widely-used exploratory technique exploited to identify groups of similar network flows.

In the methodology we propose, named SaFe-NeC, we aim at addressing the lack of parameter-free solutions in the state of the art, building upon the characteristics of supervised and unsupervised techniques by combining the strengths of both approaches. SaFe-NeC introduces innovative self-assessment features and adds new domain-driven semantics to data mining results, provides valuable insights to network experts and helps them exploit advanced data mining techniques. A self-evolving model evaluation phase is included to continuously track the quality degradation of the model itself, whose rebuilding is triggered as soon as quality indicators fall below a threshold of tolerance.

This paper is organized as follows. Section II describes the main building blocks of SaFe-NeC. Section III discusses the preliminary experiments performed to show the potential of SaFe-NeC on a real traffic dataset, while Section IV draws conclusions and presents future developments of this work.

II. Methodology

The SaFe-NeC methodology introduces a two-step approach to build a self-evolving model able to (i) autonomously identify homogeneous groups of traffic data without prior knowledge and (ii) classify new traffic flows in real time. Step (i) is based on a scalable clustering algorithm, followed by a domain-driven post-processing phase that enriches the anonymous clustering results to provide valuable semantics in the networking context. Step (ii) builds a classification model by exploiting clusters as labels. The classification model is then applied to new traffic flows by labeling them in real time. The model can be tuned to maximize both the technical performance, i.e., its accuracy, and the insights into the data, i.e., the human-readability of the model, by selecting

The research leading to these results has received funding from the European Union under the FP7 Grant Agreement n. 619633 (“ONTIC” Project).

among different classification algorithms. At each step and over time, quality indicators are used to self assess the model fitting and its results. When the quality indicators fall below given thresholds, the model is automatically rebuilt to better fit new data, thus providing self-evolutionary features. The whole methodology has been built to provide a scalable, semi-autonomous and flexible approach, able to cope with very large datasets, to offload the user from arbitrary parameter choices, and to adapt to domain-specific requirements and semantics.

The rest of this Section describes the building blocks of the proposed methodology: network measurement collection, self-learning data characterization, classification model training, real-time data labeling, and self-evolving model evaluation.

A. Network measurement collection

As a use case to show SaFe-NeC's potential, we collect and analyze TCP network measurements. To this aim, a passive probe is located on the access link (vantage point) that connects an edge network to the Internet. The passive probe sniffs all incoming and outgoing packets flowing on the link, i.e., packets directed to a node inside the network and generated by a node in the Internet, and vice versa. The probe runs Tstat [18], a passive monitoring tool allowing network and transport layer measurement collection. Tstat rebuilds each TCP connection by matching incoming and outgoing segments. Thus, a flow-level analysis can be performed [19]. A TCP flow is identified by snooping the signaling flags (SYN, FIN, RST). The status of the TCP sender is rebuilt by matching sequence numbers on data segments with the corresponding acknowledgement (ACK) numbers.

Among Tstat's many measures, we selected the following:

- the minimum ($RTTMin$) and maximum ($RTTMax$) *Round-Trip-Time* observed on a TCP flow, for both the client and the server, separately
- the *number of hops* ($Hops$) from the remote node to the vantage point observed on packets belonging to the TCP flow, as computed by reconstructing the IP Time-To-Live[1]
- the *flow reordering probability* (P_{reord}), which can be useful to distinguish different paths
- the *flow duplicate probability* (P_{dup}), that can highlight a destination served by multiple paths[2]
- the total number *of packets* ($NumPkt$), *of data packets* ($DataPkt$), and *of bytes* ($DataBytes$) sent from both the client and the server, separately
- the minimum ($WinMin$), maximum ($WinMax$), and scale ($WinScale$) values of the *TCP congestion window* for both the client and the server, separately
- the *class of service* ($Class$), as defined by Tstat, e.g., HTTP, video, VoIP, SMTP, etc.

[1]The initial TTL value is set by the source, typical values being 32, 64, 128 and 255.

[2]P_{reord} and P_{dup} are computed by observing the TCP sequence and acknowledgement numbers carried by segments of a given flow. We refer the reader to [19] for more details.

As discussed in Section III, the $Class$ feature is used only to select the TCP flows of interest and focus our analysis on a specific subset of data.

B. Self-learning data characterization

The collected network flows are the input records to the core of the SaFe-NeC approach: the self-learning data characterization block. This block consists of (i) a clustering phase and (ii) a post-processing enrichment phase. Before the clustering phase the dataset is normalized with the z-score technique [20].

Clustering is exploited to autonomously identify homogeneous groups of traffic flows without prior knowledge. Different clustering algorithms can be exploited to this aim in SaFe-NeC; however, in the current work we focus on the popular K-means algorithm [21].

K-means algorithm is effective for spherical-shaped clusters. Nevertheless, it is sensitive to the random initialization of centroids and to cluster size, densities of data points, non-globular shapes of clusters and outliers. K-means also requires the a-priori knowledge of the number of clusters. Despite all these limitations, K-means has seen a widespread usage for cluster analysis in many different application contexts (e.g., medical records [22], data partitioning in a computer networking [23]), for its low computational cost and the tightness of the globular clusters resulting.

SaFe-NeC's self-assessment of clustering results also provides the benefit of choosing k in K-means: clustering is performed for a wide range of k values, then the trend of quality measures, such as SSE (Sum of Squared Errors), are exploited to infer the optimal value of k. To this aim, the quality measure chosen (e.g., SSE) is plotted against k to reproduce the well-known "Elbow" graph (or "Knee" approach) ([24]): the optimal value of k must be selected at the coordinates where the gain from adding a centroid is negligible, or in other words the SSE reduction is not interesting anymore. Such procedure can be automatized, parametrizing the threshold below which an SSE reduction is not interesting; however, the domain expert can here select a compromise between the loss of information and the need to inspect a high number of clusters (see Section III-A for an example).

To make anonymous groups, as clusters are, human-inspectable, the SaFe-NeC methodology is designed to enrich the clusters with both general attribute-based statistics, and domain-specific knowledge. In the current SaFe-NeC evaluation, we associate each cluster with the number of records it contains and four attributes of interest. The number of records provides insights into the data distribution, by identifying clusters covering most of the dataset and others identifying small "remote" groups of data points. For instance, some network datasets present a predominant cluster with regular traffic and many smaller clusters identifying deviations. Other datasets may present similarly-sized clusters for different subnets or services. To this we add the 2 attributes with the lowest absolute variance and the 2 with the highest variance reduction ratio with respect to their variance over the whole normalized dataset. Given an estimator for the variance $\hat{\sigma}^2$, the variance reduction ratio (VRR) for the j-th cluster and i-th feature x^i

is defined as follows.

$$VRR_j(x^i) = \frac{\hat{\sigma}_D^2(x^i) - \hat{\sigma}_j^2(x^i)}{\hat{\sigma}_D^2(x^i)} \tag{1}$$

where $\hat{\sigma}_D^2$ is the variance over the whole dataset and $\hat{\sigma}_j^2$ is the variance over the j-th cluster.

The rationale behind the variance reduction is to quantify the information gain, for a given attribute, obtained by isolating some of the records in a cluster; it is inherited from decision trees [25], where the order in which the attributes are considered widely affects the performances and the results. Together with the variance itself, is a strong indicator of the features that characterize a cluster the most and their relative importance.

C. Classification model training

The flows processed by the clustering algorithm are labelled accordingly to the cluster id and form a labeled dataset, which can be exploited for supervised learning. The goal of this phase is thus building a classifier to efficiently label new unseen records as they are captured. In the present work, the classification algorithm is a decision tree [26], which is a popular technique able to reach both good results and easy model interpretability.The output tree provides an easy-to-read overview of the features that best split the dataset according to the labels: for each node of the tree the split criterion can be written as an if/else condition over a single feature and a splitting value, and few levels of the tree are usually sufficient to show the most significant splits for the purpose of the classification.

D. Real-time data labeling

The decision tree trained at the previous step is exploited to label new unseen data as they are captured, in a real-time fashion. The prediction algorithm relies simply on the visit of the tree, according to the rules at the nodes. The time required to label each record depends on the depth of the tree, i.e. on the number of comparisons to be performed before reaching the outcome label.

E. Self-evolving model evaluation

The quality of the trained model is subject to ageing, as the scenario identified by the clustering continuously evolves. Therefore, SaFe-NeC continuously evaluates the quality degradation of the model itself, and triggers its regeneration as soon as quality indexes fall below a threshold of tolerance. To this aim, SaFe-NeC currently exploits the SSE, which is updated every N new records of data, where N is set by the user. In the case of SSE we expect it to raise while new data are added to the model (see Section III-C for an example). As soon as the SSE raises above a given threshold, expressed as a percentage variation from the initial SSE, the entire process must restart from the clustering phase, either from scratch or in an incremental way, if supported by the selected clustering algorithm. The new clustering can be executed on the whole historical dataset or on the most recent flows only; the latter option generates a more specific up-to-date model, that could be less general due to the fewer training data.

TABLE I. CLUSTERS' CHARACTERIZATION

id	num. records	Top-2 representative attributes: Ranking by highest variance reduction ratio		Ranking by lowest absolute variance	
		Attribute	Avg. value	Attribute	Avg. value
0	247053	P_{reord}	4.06E-6	$DataBytes_S$	225654
		$WinScale_S$	3	$DataBytes_C$	189340
1	38	$DataBytes_C$	506	$DataBytes_C$	506
		$DataPkt_C$	3	$WinMax_S$	73726
2	52	$DataBytes_C$	577	$DataBytes_C$	577
		$DataPkt_C$	4	$DataBytes_S$	158772
3	105	$Hops$	12	$WinMax_S$	122511
		$WinMax_S$	122511	$DataBytes_S$	5946043
4	96	$DataBytes_C$	436	$DataBytes_C$	436
		$DataPkt_C$	2	$DataBytes_S$	60723
5	119	$DataBytes_C$	424	$DataBytes_C$	424
		$DataPkt_C$	3	$DataBytes_S$	9487
6	62	$DataBytes_C$	2924	$DataBytes_C$	2924
		$DataPkt_C$	17	$WinMax_S$	93271
7	55	$DataBytes_C$	1085	$DataBytes_C$	1085
		$DataPkt_C$	7	$WinMax_S$	53991

III. PRELIMINARY EXPERIMENTAL VALIDATION

A set of preliminary experimental results have been obtained on a real traffic network dataset. The performed experiments address (i) parameter setting analysis, (ii) data cluster characterization through the proposed measures (e.g., variance reduction ratio and absolute variance), (iii) classification algorithm evaluation, and (iv) self-evolving model evaluation.

The dataset used for the experimental validation consists of 1 million flows as recorded by Tstat, extracted from the logs of a backbone-link probe of a nation-wide ISP in Italy with 3 different vantage points. ISP vantage points expose traffic of 3 different Points-of-Presence (POP) in different cities of Italy, each of them aggregating traffic from more than 10,000 ISP customers. Such dataset counts for approximately 79.6 hours of flows, that correspond to an average of 3.48 complete flows per second. The preprocessing part of the algorithm filters out the HTTP and HTTPS flows to focus the analysis on more heterogeneous data, resulting in a dataset of approximately 500,000 records. The first half of this dataset was then reserved to the training phase, while the second half was kept for the simulation of the real-time arrival of new records.

SaFe-NeC has been developed in Scala by exploiting the k-means and decision-tree algorithms available in MLLib [27]. Experiments were performed on a cluster of 5 nodes, each with a 2.67 GHz six-core Intel(R) Xeon(R) X5650 and 32 GB RAM, running Cloudera Distribution of Hadoop (CDH5.3.1).

A. Parameter setup

Figure 1 shows the SSE plot for a number of different k. The candidates for k that can be chosen following the method explained in Section II-B are 4 and 8. The latter especially does not see any improvement when an extra centroid is added, and is therefore our choice for this test. However, a network analyst could choose $k = 4$ for the ease of analysis or other background knowledge.

B. Data characterization

The results of a K-means run for $k = 8$ are shown in Table I. As we can see, cluster 0 absorbs most of the records, while the remainder is scattered among the others. To characterize this anonymous piece of knowledge we inspect

TABLE II. Statistics of a representative cluster (Cluster 0 in Table I). Top-3 attributes for each ranking

Ranking by highest variance reduction ratio				Ranking by lowest absolute variance			
Attribute	Variance reduction ratio	Average value	Variance	Attribute	Variance reduction ratio	Average value	Variance
P_{reord}	0.9995	4.06E-06	1.01E-04	$DataBytes_S$	-1.46E-03	225654	1.18E-19
$WinScale_S$	1.88E-04	3	2.19E-07	$DataBytes_C$	-1.73E-03	189340	4.90E-19
$WinMin_S$	1.19E-04	23071	4.61E-15	$WinMax_S$	-4.09E-04	143591	6.52E-19

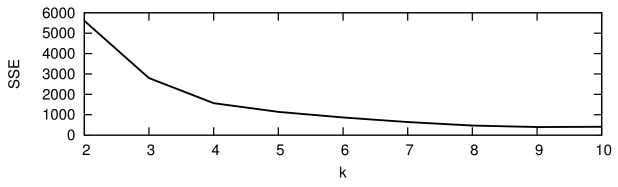

Fig. 1. SSE with respect to the number of clusters k

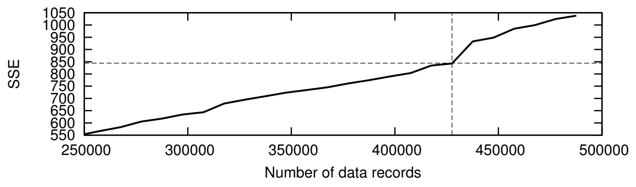

Fig. 2. Model quality indicator (SSE) trend as new fresh data are automatically labeled by the classifier

the two most representative attributes, as discussed in Section II-B. The ranking by increasing variance shows us that the most significant attributes are most often $DataBytes_S$ and $DataBytes_C$, the total number of bytes sent by server and client respectively. $DataBytes_C$ is as well ranked first by VRR in 6 of the 8 clusters, which confirms its sensitivity towards the clustering results. However, with the mere ranking by absolute variance we do not have many clues on a set of features distinguishing between the clusters. The VRR highlights instead a feature that seems to clearly depict what has gathered the records into cluster 0: a P_{reord} close to 0. Such consideration can lead to interpret cluster 0 as a prototype of ideality whereas the other clusters can be seen as outliers or anomalies. Many of them are characterized by the amount of data transferred from the client, and in particular some have an average of very few data packets per flow. An exception to this trend is cluster 3, which appears to be clustered mostly for $Hops$. We exploit all these features to provide a synthetic description for each cluster; such overview can be a reliable tool for an analyst to begin its investigations on the QoS or detect anomalies in the network.

Table II shows a summary for cluster 0, for the top three attributes characterizing the cluster. We recall that the clustering and the rankings are computed on the normalized values, while this table shows the actual average and variance, to provide a sensible value. The table clearly shows that the variance for P_{reord} has dropped by more than 3 orders of magnitude; the VRR for the second feature ranked, $WinScale_S$, is negligible, especially as the variance itself was negligible for this feature. The first-ranked feature is thus a sufficient description of the cluster. The ranking by absolute variance shows instead three features with infinitesimal variance, which has had a slight but still negligible augment in all the three cases. In this cluster, such features thus seem to provide more a general depiction of the data than a specific characteristic of the cluster itself; however, it is still a useful insight for the analyst.

C. Model training and real-time labeling

We built a decision tree with a maximal depth of 4, using Gini impurity as measure of the information gain. The validation was done using a 3-fold cross-validation scheme on the training data.The average accuracy on the three cross validation runs is 99.6%. The results for precision and recall for each class are not shown here due to the lack of space; they are overall encouraging, with only two classes below the 75% of precision and 80% of recall. Then, a tree generated on the whole training dataset has been applied to new records (i.e., the second half of the original dataset), to simulate the real-time arrival of fresh data. To show the SSE evolution, we did not trigger any model rebuilding. Figure 2 shows the evolution of the SSE as new records are labeled by the decision tree. The value of SSE steadily increases, as expected, as any new record contributes to it. Interestingly, a sudden raise just before the 430000th flow is present, a hint that could be a symptom of a rapid evolution of the modeled state of the network, rather than due to the natural imprecision of the model. If the threshold for recomputing the model had been set at a 50% increase of the SSE with respect to the starting value, on our dataset, it would have been triggered after the 467570th record. Supposing to trigger, then, the model rebuilding, the new SSE curve for choosing K, not reported here, would show that the new best value is $K = 11$, with the SSE dropping at 335.9. With K=11, new clusters will appear in the next run, and the whole characterization process will tag them based on their most specific features. The prediction of a new label takes an average time of 2.18ms, including the time to update the SSE and store the new labeled dataset.

IV. Conclusion

This paper presents the SaFe-NeC methodology for a self-learning data analytics system capable of effectively mine network traffic datasets. SaFe-NeC exploits a two-step approach to build a self-evolving model to derive semantically-enriched and homogeneous groups of traffic data without prior knowledge and classify new traffic flows in real time. SaFe-NeC also includes self-evolutionary features to automatically rebuild the network traffic model to better fit new data. To show the capabilities of the methodology, we performed experiments on a large dataset of TCP flows statistics, but the whole approach can be easily applied to any kind of data, e.g. headers, flows and statistical measurements.

We expect to successfully extend the proposed methodology by introducing new comprehensive quality indicators, such as the Silhouette, and evaluating advanced clustering algorithms such as DBScan[28], able to semi-autonomously recognize outliers and noisy points.

References

[1] Y. Lee and Y. Lee, "Toward scalable internet traffic measurement and analysis with hadoop," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 1, pp. 5–13, Jan. 2012.

[2] D. Apiletti, E. Baralis, T. Cerquitelli, S. Chiusano, and L. Grimaudo, "Searum: A cloud-based service for association rule mining," in *11th IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA-13*, 2013, pp. 1283–1290.

[3] A. Bär, A. Finamore, P. Casas, L. Golab, and M. Mellia, "Large-scale network traffic monitoring with dbstream, a system for rolling big data analysis," in *2014 IEEE International Conference on Big Data, Big Data 2014, Washington, DC, USA, October 27-30, 2014*, 2014, pp. 165–170.

[4] E. Baralis, L. Cagliero, T. Cerquitelli, S. Chiusano, P. Garza, L. Grimaudo, and F. Pulvirenti, "Nemico: Mining network data through cloud-based data mining techniques," in *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, ser. UCC '14, 2014, pp. 503–504.

[5] D. Apiletti, E. Baralis, T. Cerquitelli, and V. D'Elia, "Characterizing network traffic by means of the netmine framework," *Computer Networks*, vol. 53, no. 6, pp. 774–789, 2009.

[6] M. Hossain, S. Bridges, and R. Vaughn Jr, "Adaptive intrusion detection with data mining," *IEEE Internation Conference on Systems, Man and Cybernetics*, vol. 4, 2003.

[7] F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb, "Minerals: using data mining to detect router misconfigurations," in *MineNet '06*. New York, NY, USA: ACM Press, 2006, pp. 293–298.

[8] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: multilevel traffic classification in the dark." in *SIGCOMM*, 2005, pp. 229–240.

[9] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *SIGMETRICS '05*. New York, NY, USA: ACM Press, 2005, pp. 50–60.

[10] Y. Ouyang, M. H. Fallah, S. Hu, Y. R. Yong, Y. Hu, Z. Lai, M. Guan, and W. Lu, "A novel methodology of data analytics and modeling to evaluate LTE network performance," in *2014 Wireless Telecommunications Symposium, WTS 2014, Washington, DC, USA, April 9-11, 2014*, 2014, pp. 1–10.

[11] M. Amer, M. Goldstein, and S. Abdennadher, "Enhancing one-class support vector machines for unsupervised anomaly detection," in *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, ser. ODD '13, 2013, pp. 8–15.

[12] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion detection with unlabeled data using clustering," *Proceedings of ACM CSS Workshop on Data Mining Applied to Security, PA,, November*, 2001.

[13] J. Erman, M. Arlitt, and A. Mahanti, "Traffic classification using clustering algorithms," in *MineNet '06*. New York, NY, USA: ACM Press, 2006, pp. 281–286.

[14] Q. Wang and V. Megalooikonomu, "A clustering algorithm for intrusion detection," *Proc. SPIE*, vol. 5812, pp. 31–38, 2005.

[15] E. Baralis, A. Bianco, T. Cerquitelli, L. Chiaraviglio, and M. Mellia, "Netcluster: A clustering-based framework to analyze internet passive measurements data," *Computer Networks*, vol. 57, no. 17, pp. 3300–3315, 2013.

[16] L. Grimaudo, M. Mellia, E. Baralis, and R. Keralapura, "Select: Self-learning classifier for internet traffic," *IEEE Transactions on Network and Service Management*, vol. 11, no. 2, pp. 144–157, 2014.

[17] D. Giordano, S. Traverso, L. Grimaudo, M. Mellia, E. Baralis, A. Tongaonkar, and S. Saha, "Youlighter: An unsupervised methodology to unveil youtube CDN changes," *CoRR*, vol. abs/1503.05426, 2015.

[18] A. Finamore, M. Mellia, M. Meo, M. Munafò, and D. Rossi, "Experiences of internet traffic monitoring with tstat," *IEEE Network*, vol. 25, no. 3, pp. 8–14, 2011.

[19] M. Mellia, M. Meo, L. Muscariello, and D. Rossi, "Passive analysis of tcp anomalies," *Computer Networks*, vol. 52, no. 14, pp. 2663–2676, 2008.

[20] J. Han, *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005.

[21] B.-H. Juang and L. Rabiner, "The segmental k-means algorithm for estimating parameters of hidden markov models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, no. 9, pp. 1639–1641, Sep 1990.

[22] A. L. Buczak, L. J. Moniz, B. H. Feighner, and J. S. Lombardo, "Mining electronic medical records for patient care patterns," in *IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, 2009, pp. 146–153.

[23] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. New York, NY, USA: ACM, 2005, pp. 593–599.

[24] Pang-Ning T. and Steinbach M. and Kumar V., *Introduction to Data Mining*. Addison-Wesley, 2006.

[25] L. Rokach and O. Maimon, *Data Mining with Decision Trees: Theroy and Applications*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2008.

[26] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[27] A. Spark, "The Apache Spark scalable machine learning library. Available: https://spark.apache.org/mllib/," 2013.

[28] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, 1996, pp. 226–231.