Anomaly Detection Algorithms for the Sleeping Cell Detection in LTE Networks

Sergey Chernov, Michael Cochez and Tapani Ristaniemi
Department of Mathematical Information Technology
University of Jyväskylä, Finland
sergey.chernov@jyu.fi michael.cochez@jyu.fi tapani.ristaniemi@jyu.fi

Abstract—**The Sleeping Cell problem is a particular type of cell degradation in Long-Term Evolution (LTE) networks. In practice such cell outage leads to the lack of network service and sometimes it can be revealed only after multiple user complains by an operator. In this study a cell becomes sleeping because of a Random Access Channel (RACH) failure, which may happen due to software or hardware problems. For the detection of malfunctioning cells, we introduce a data mining based framework. In its core is the analysis of event sequences reported by a User Equipment (UE) to a serving Base Station (BS). The crucial element of the developed framework is an anomaly detection algorithm. We compare performances of distance, centroid distance and probabilistic based methods, using Receiver Operating Characteristic (ROC) and Precision-Recall curves. Moreover, the theoretical comparison of the methods' computational efficiencies is provided. The sleeping cell detection framework is verified by means of a dynamic LTE system simulator, using Minimization of Drive Testing (MDT) functionality. It is shown that the sleeping cell can be pinpointed.**

I. Introduction

Ever increasing user demands to wireless communication services and the inevitable growth of mobile cellular network complexity pose ambitious challenges for mobile operators and the telecommunication research community. The high popularity of tablet computers and smartphones has led to a fierce competition among wireless network providers. To attract customers, the operators are forced to offer high quality services and more network traffic for less money. On the other hand, the configuration and maintenance of network equipment is a rather expensive process, which in many cases requires manual work of qualified specialists. One of the reasons is the heterogeneous nature of wireless networks, i.e. they combine different radio access technologies (HSPA, OFDM) and cell layers (macro, micro, pico). In order to cope with the high level of network complexity and to avoid an increase in capital expenditures, 3GPP has been developing the Self-Organizing Network (SON) concept since Release 8 [1].

The Self-Organizing Network concept introduced by 3GPP consists of three solutions, namely Self-Configuration, Self-Optimization, and Self-Healing [2]. Self-Configuration refers to the automated configuration of the newly established networks, while Self-Optimization includes changing the network parameters in order to meet the current demands on the network. The purpose of the Self-Healing mechanism is to detect and address problems automatically, avoiding significant impact on subscribers' experience and reducing operational expenses. In one of the earliest research efforts [3] neighbor cell list reports are considered in order to evaluate the network functionality. Although, the algorithm has shown good performance in failure detection, there is no possibility to detect problems with a low number of active users. In a recent work the authors of [4] propose anomaly detection and a diagnosis framework for mobile network operators.

Our current study is related to the sleeping cell detection problem caused by RACH failure. RACH failures can occur when there is no radio coverage outage. Therefore, instead of using radio environment measurements, we analyze sequences of events reported by Mobile Terminals (MTs). In [5] we proposed a data mining approach that enables automatic detection of malfunctioning cells. In a further study [6] we considered the effect of the size of the N-grams for the feature selection algorithm. Based on that research, we concluded that the influence of the size is minimal, and hence we choose N-grams of size 1 in this article.

In the current article we focus on another part of the designed data mining structure. The quality of the sleeping cell detection depends on the utilized data mining algorithm. Therefore, we compare, against each other, the application of different classification methods in our cell outage detection framework. The methods used are based on distance (k-Nearest Neighbors, k-NN), centroid distance (Self-Organizing Map, SOM), and probabilistic data structures (Local-Sensitive Hashing, LSH and Probabilistic Anomaly Detection, PAD). The practical comparison involves analysis of ROC and Precision-Recall curves. Finally, we present theoretical bounds of computational complexities of the considered algorithms.

The rest of the paper is organized as follows. We provide the description and simulation details of the sleeping cell detection problem and RACH failure as its particular case in Section 2. A short overview of the utilized simulation tool and related assumptions is given in the same section. Afterwards, the cell outage detection framework and its functional parts are presented in Section 3. Section 4 demonstrates the main results and compares performances of the considered anomaly detection methods. Finally, Section 5 concludes the paper.

II. Sleeping Cell Problem and Simulation

A. Sleeping Cell Problem

Sleeping cell is a situation whereby a base station's malfunctioning is not detected by the operator as there is no alarm triggered. There could be multiple reasons for this to happen, including misconfiguration, excessive load or software/firmware problem at the base station side. The main danger is that the failure remains invisible for the network maintainers and would be revealed only after a number of complaints from subscribers, in other words unsatisfied customers. There are three types of sleeping cells. *Impaired* and *crippled* cells show, respectively, slightly and significantly lower performances than expected. The most critical type is a *catatonic* cell, which leads to the complete absence of service.

In our study we consider sleeping cell caused by RACH failure. This type of failure makes it impossible for users to establish a connection or to make a handover to the malfunctioning cell. However, MTs previously connected to the sleeping cell do not experience any lack of service. Hence, the cell starts out being impaired since only few users notice problems, but over time it becomes catatonic since it does not serve any MTs. Typically, a RACH problem would be detected only after long observation time or after multiple users complaints. Thus, a timely detection of RACH failures is an important issue in LTE networks.

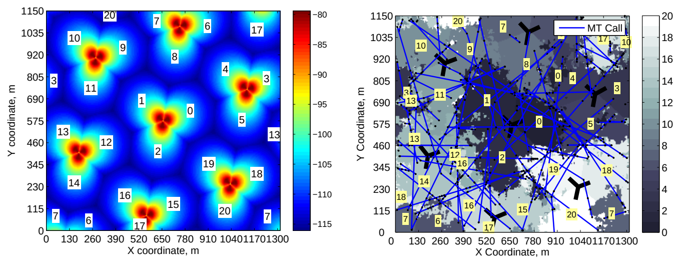
B. Simulation

The simulator used in our experiments has been designed in accordance with LTE 3GPP specifications and is utilized by the Nokia Network research group. The data are MDT measurements generated using an advanced LTE system level simulator.

The simulations have been run several times for a duration of 572 simulated seconds each. The first simulation was conducted without cell outages and corresponds to the normal operation of the network. The normal patterns of the MDT logs are learned and captured by the data mining framework. Cell 1 happened to have a RACH failure in the beginning of the second run. In spite of the failure, Cell 1 was able to provide services to the connected MTs. However, it finally became a *catatonic* sleeping cell.

The utilized scenarios consist of 7 sites, respectively 21 cells, with a BS distance of 500 meters, see fig. 1(a). The model of the radio propagation environment includes slow and fast fading conditions. MTs were initialized uniformly random on the map and traveled under a random walk mobility model. Table I contains the whole list of the simulation parameters.

The simulations provided us with two kinds output: MDT logs and signal strength information. MDT logs are described by timely ordered sequences of variables, but we have employed MDT triggering events (listed in table II), MT IDs and target cell IDs. Signal strength information reports the distribution of the cells' signal strengths on the whole map. Hence, the dominance areas of the cells can be drawn, see fig. 1(b).



(a) ***Network map. SINR*** **(b)** ***Mobile terminal calls on slow faded dominance map***

Fig. 1. Wrap around Macro 21 scenario

TABLE I. General Simulation Parameters

Parameter	Value
Cellular layout	Macro 21 wrap-around
Number of cells	21
UEs per cell	15
Inter-Site Distance	500 m
Link direction	Downlink
User distribution in the network	Uniform
Maximum BS TX power	46 dBm
Initial cell selection criterion	Strongest RSRP value
Simulation length	572 s ($\approx$ 9.5 min)
Simulation resolution	1 time step = 71.43 μs
Max number of UEs/cell	20
UE velocity	30 km/h
Duration of calls	Uniform 30 to 140 s
Traffic model	Constant Bit Rate 320 kbps

TABLE II. Network events triggering MDT log entry

A2 RSRP ENTER — RSRP goes under A2 enter threshold.
A2 RSRP LEAVE — RSRP goes over A2 leave threshold.
A3 RSRP — A3 event, according to 3GPP specification.
HO COMMAND RECEIVED — handover command received [7].
HO COMPLETE RECEIVED — handover complete received [7].
PL PROBLEM — Physical Layer Problem [8].
RLF — Radio Link Failure [7].

III. Cell Outage Detection Framework

The whole sleeping cell detection framework involves two separate procedures, see fig. 2. At first, the model learns a "normal" network behavior. It practically means that the footprint of normal sub-calls in a feature space and necessary thresholds are extracted. Afterwards, the current network state is compared against the previously trained model, and cell outage predictions are made.

In contrast to our previous work [6] we have not employed dimensionality reduction. Actually, there is no high need for it, since data are fully described by seven 1-gram features.

In the current research we focus on the choice of an anomaly detection algorithm as the main functional block of the scheme. The utilized methods are based on distance (k-NN), centroid distance (SOM), and probabilistic data structures (LSH, PAD). The practical comparison is carried out by means of ROC and Precision-Recall curves. The description of the other functional parts is provided in the following sub-sections.

A. Sliding Window

Each MT causes an ordered sequence of events in the LTE network. Due to the random nature of calls, the sequences have different lengths. In order to do length-wise normalization and localize calls, we slice the sequences to sub-sequences, or sub-calls, by means of a sliding window. The chosen window step and size allow a sub-call to visit three cells on average.

B. N-Gram Analysis

In order to extract features from sub-calls, which are sequential data, we make use of N-gram analysis. An N-gram is an ordered set of N terms. The number of N-grams

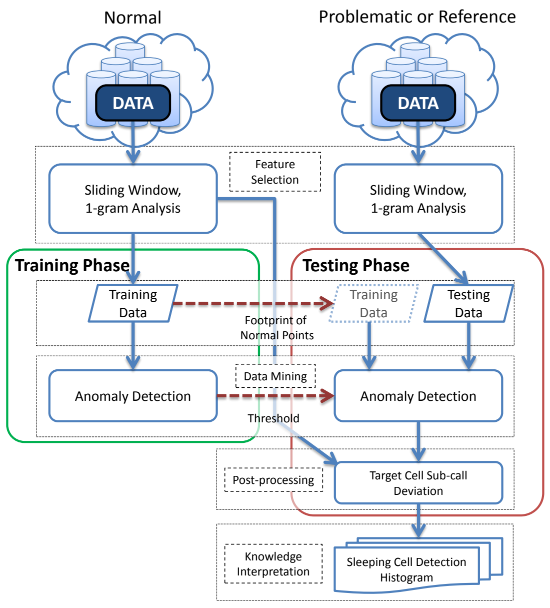


Fig. 2. Sleeping cell detection framework

is defined as all possible permutations of unique instances in the considered data. In our research, the instances are network events, see table II. Thus, sub-calls can be transformed into feature vectors, which indicate how often each particular N-gram occurs in these subsequences.

C. Anomaly Detection Algorithms

We define *abnormal* sub-call as sub-call that moved into or travelled through a sleeping cell, otherwise the sub-call is referred as *normal*. An *outlier* sub-call is a sub-call detected as abnormal and a *non-outlier* sub-call is one detected as normal. Next, we introduce binary classification methods, which split the sub-calls into the *outlier* and *non-outlier* classes.

1) k-NN anomaly score outlier detection algorithm: this algorithm belongs to the family of distance based anomaly detection methods. The sum of distances to the k nearest neighbors of a point is referred to as k-NN anomaly score of that point [9]. In practice, k is usually chosen to be squared root of the number of points in the dataset.

The higher the k-NN anomaly score the more chances for a point to be assigned to the *outlier* class. The exact decision boundary is introduced as $n\%$ quantile of normal dataset. Thus, the defined threshold splits the considered dataset into two classes: *non-outlier* class, having low anomaly scores, and *outlier* class, having high anomaly scores.

2) Self-organizing map: SOM is a set of connected nodes which “memorizes” the shape of a larger set of points [10]. SOM can be used for binary classification problems by considering points which are located in the interior of the map being member of the one class and points which fall outside it in the other.

For a given set of points, in our case the set of normal sub-calls, the map is created by morphing a set of interconnected nodes such that it covers the points. The morphing is an iterative process and happens by moving the best matching unit, i.e. closest node of the map, closer towards given random training point. This also affects the connected nodes to a smaller extend. We utilized a two dimensional, linearly initialized self-organizing map with a hexagonal lattice. [1] SOM anomaly scores are assigned by mean of k-NN.

3) Probabilistic anomaly detection: PAD calculates density functions of features of the normal dataset [11]. The untypical or abnormal behavior can be defined as an event with a small probability. There are two types of exploited density functions, referred to as consistency checks. First and second order consistency checks estimate, respectively, unconditional and conditional probabilities of elements. The Friedman-Singer estimator was used for the probability calculations. It allowed us to estimate the probabilities for already known as well as previously unknown elements.

During the training phase, PAD is fed by data representing normal network behavior, and consistency checks are calculated. A testing record is labeled as abnormal if it has not passed one of the consistency checks, i.e. the probability of one of its elements is lower than a predefined threshold.

4) Locality-sensitive hashing: LSH first introduced as a method for finding approximate nearest neighbors, given a distance measure, d, and a threshold for the error, ϵ [12]. In order to use the method, one needs a family of hash functions, which likely map elements with a high similarity to the same value and elements with low similarity to different ones. For our experiment, the Jaccard distance measure ($d\left(A,B\right) = 1 - sim\left(A,B\right)$) and its associated family of locality sensitive hash functions, namely min-hash, were used.

Concrete, to test whether a sub-call is abnormal, we use LSH and try to find near neighbors in our training data set. If these are not found, we conclude that the sub-call is abnormal.

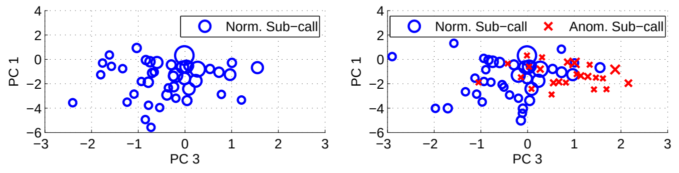
D. ROC and Precision-Recall curves

Common ways to represent the performance of binary classifiers are ROC and Precision-Recall curves.

- *Precision* is the number of *abnormal* sub-calls, classified as *outliers*, divided by the total number of *outlier* sub-calls.
- *Recall* or *True Positive Rate* (TPR) is the number of *abnormal* sub-calls, classified as *outliers*, divided by the number of *abnormal* sub-calls.
- *False positive rate* (FPR) is the number of *normal* sub-calls, classified as *outlier*, divided by the number of *normal* sub-calls.

ROC curve represents TPR in function of FPR, while Precision-Recall shows the relation between precision and recall. A better algorithm will have a higher precision and recall and a lower FPR. We used 6-fold cross validation and included information about the standard deviation in the curves. The ROC curves also contain the operating point which shows the actual algorithm’s performance. This point corresponds to the threshold used to split the sub-calls.

[1] We used the SOM Toolbox from http://www.cis.hut.fi/projects/somtoolbox/



(a) ***Normal dataset*** **(b)** ***Problem dataset***

Fig. 3. Data in the feature space. Marker size represents the density of points

E. Post-processing

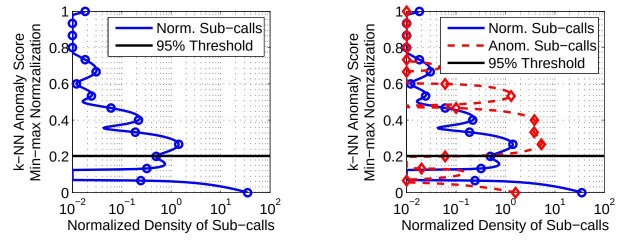
Post-processing is implemented by the *target cell deviation of sub-calls* algorithm, which makes use of target cell feature of sub-calls. Two groups of sub-calls are taken into account. The first one consists of all instances of the training dataset. The second one is composed of the sub-calls assigned to the *outlier* class by the anomaly detection algorithm during the testing phase. Next, two histograms representing the distribution of target cell feature among cells are calculated and normalized by the corresponding number of sub-calls. The deviation histogram is obtained as the absolute difference between testing and training histograms.

The deviation histogram is not the final evaluation of the network performance. A sub-call in itself hits several cells. Hence, the *outlier* sub-calls can indicate as malfunctioning not only the actually sleeping cell, but also its neighbors. The proposed amplification procedure increases a sleeping cell's bar on the deviation histogram, while decreasing the neighboring bars. It is worth to point out that if there is no cell outage then the deviation histogram is not significantly changed by the amplification. Finally, the amplified deviation histogram is normalized and introduced as sleeping cell detection histogram, see fig. 6.

In accordance to our detection framework the ordered event sequences generated by MTs are sliced into pieces by sliding window with size 15 and step 10. 1-gram analysis construct the set of features, which are basically network events listed in table II. Normal and problem datasets projected on two principal components of the problem dataset is plotted in fig. 3. This first analysis of the data set shows that it is not trivial to separate the *abnormal* sub-calls from *normal* ones.

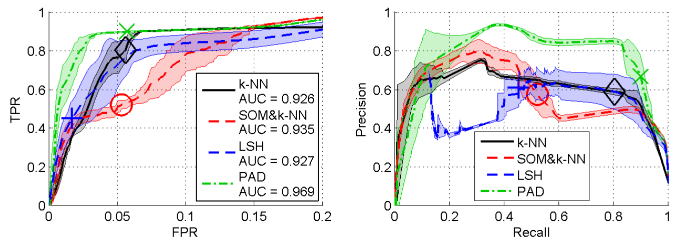
An anomaly detection algorithm both assigns anomaly scores to dataset's points and separates them into *outlier* and *non-outlier* classes. In fig. 4 we plotted the normalized density of sub-calls in function of the k-NN anomaly score. Note that in the figures we swapped the X and Y axis and that the density is plotted on a logarithmic scale. A black line, located around anomaly score 0.2, separates the figure in two areas. This decision threshold is determined during the training phase, such that the anomaly scores below the line represent 95% of the points in the normal dataset, while the remaining 5% are located above the line. The chosen threshold is mapped to ROC and Precision-Recall curves as an operating point.

Figure 4(b) shows how the decision threshold works on *i*) *normal* sub-calls *ii*) *anomalous* sub-calls from the problem dataset. Note that many of the anomalous points obtain a high anomaly score and will hence be correctly classified as *outliers*. The points of curve of *anomalous* sub-calls under the threshold



(a) ***Normal dataset*** **(b)** ***Problem dataset***

Fig. 4. Min-max normalization of k-NN anomaly scores



(a) ***ROC curves*** **(b)** ***Precision-Recall curves***

Fig. 5. Performances of anomaly detection algorithms. Colored areas show 3 standard deviations of the lines. Makers represent operating points: "◇" — k-NN, "◯" — SOM&k-NN, "×" — PAD, "+" — LSH

indicate sub-calls which will be erroneously classified as *non-outliers*. The figure also shows the source of false positives. These are sub-calls which are represented by the normal curve above the threshold. Similar figures can be plotted for the other anomaly detection algorithms, but we left them out for brevity.

We compare the performance of the different anomaly detection methods visually using ROC curves in fig. 5(a) and Precision-Recall curves in fig. 5(b). Note that we show only the most interesting part of the ROC curves. Points with a FPR greater than 0.2 result in pretty much the same TPR for all methods. For interpretation of these figures, we remind that an optimal method would find a TPR of 1 and a FPR of 0 and hence the ROC curve would have a point in the top-left corner of the chart. For the Precision-Recall curve the optimal method would find a precision and recall equal to one corresponding to a point in the top-right corner.

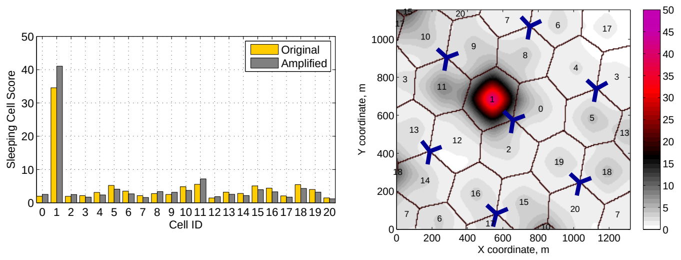
From the curves, we see that the PAD method performs better as the other methods. SOM&k-NN, despite the fact that it does not have the smallest AUC, could be considered the weakest method. In order to get the TPR at an acceptable level, one has to accept a high false positive rate. k-NN as such scores fairly strong. Its main benefit is that is very stable which can be seen from the small standard deviations. The LSH method scores similar to k-NN in the regions of interest, i.e. these with high precision and recall. However, the method seems somewhat unstable. In other words, when performing analysis using LSH on a limited dataset, the result might be unreliable.

Another consideration regarding the used anomaly detection methods is their computational complexity. In a real-life scenario the resource budget allocated for a Self-Healing algorithm might be of great importance. In table III we listed the computational complexities of the training and testing

TABLE III. COMPUTATIONAL COMPLEXITIES OF ANOMALY DETECTON ALGORITHMS

Algrithm	Training phase	Testing phase
k-NN [13]	$O(dN_{trn}^2)$	$O(dN_{trn}N_{tst})$
SOM & k-NN [14]	$O(duN_{trn})$	$O(d\sqrt{u}N_{tst})$
LSH	$O(brN_{trn})$	$O(brN_{tst})$
PAD [15]	$O(v^2N_{trn}^2)$	$O(N_{tst})$

N_{trn}, N_{tst} — number of instances in the training and testing dataset respectively
d — number of N-gram features or dimensions
u — number of unique instances in N_{trn}
v — number of unique record values for each record component
b and r — number of bands and rows used for LSH



(a) ***Sleeping cell detection histogram*** **(b)** ***Heat map***

Fig. 6. Sleeping cell detection

phases. As we can see, LSH is the only method which, with fixed values for b and r, remains linear for both phases. Other methods show a roughly quadratic complexity during training. During testing, k-NN loses out with again a quadratic complexity compared to a linear complexity for PAD. SOM&k-NN outperforms k-NN and shows slightly better result than during its training. The PAD method seems to utilize the least computation resources to label testing dataset.

Figure 6 shows the anomaly scores obtained using k-NN anomaly detection method. Note that the bars at Cell 1 are significantly higher as what is found for the other cells. From the histogram it is obvious that we are able to detect the sleeping cell correctly.

Even though we observe differences in the anomaly detection methods, we note that all of them were able to determine the sleeping cell correctly. Hence, we find that the performance of the overall detection framework is not strongly dependent on the choice of the classifier. Likely, the post-processing method cancels out the small differences between the considered anomaly detection methods.

IV. CONCLUSION

In this paper we introduced a data mining framework for sleeping cell detection, caused by RACH failure. One of the crucial parts of the framework is the anomaly detection algorithm. We compared computational complexity as well as practical performances of several algorithms based on a simulation. The considered algorithms have shown different detection rates on the simulated scenario. PAD showed the best result for the detection of abnormal sub-calls, but it is computationally expensive to train. Finally, we noticed that the utilized post-processing algorithm mitigated the differences between methods' performances.

ACKNOWLEDGMENT

The authors would like to thank Fedor Chernogorov from Magister Solutions Ltd. for providing the simulation data. Further, the work is made financially possible by the Graduate School in Electronics, Telecommunications and Automation.

REFERENCES

[1] 3GPP, "Telecommunication management; Self-Organizing Networks (SON); Concepts and requirements," 3rd Generation Partnership Project (3GPP), TS 32.500, Sep. 2014.
[2] S. Hämäläinen, *LTE Self-Organising Networks (SON)*. Wiley,, 2012.
[3] C. M. Mueller, M. Kaschub, C. Blankenhorn, and S. Wanke, "A cell outage detection algorithm using neighbor cell list reports," in *IWSOS*. Springer-Verlag, 2008, pp. 218–229.
[4] S. Nováczki, "An improved anomaly detection and diagnosis framework for mobile network operators," in *DRCN'13*, 2013, pp. 234–241.
[5] F. Chernogorov, T. Ristaniemi, K. Brigatti, and S. Chernov, "N-gram analysis for sleeping cell detection in lte networks." in *ICASSP*. IEEE, 2013, pp. 4439–4443.
[6] S. Chernov, F. Chernogorov, D. Petrov, and T. Ristaniemi, "Data mining framework for random access failure detection in. lte networks." in *PIMRC*. IEEE, 2015.
[7] S. Sesia, M. Baker, and I. Toufik, *"LTE - The UMTS Long Term Evolution: From Theory to Practice"*. John Wiley & Sons, 2011.
[8] *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification (Release 10)*, 3GPP Std. TS 36.331, 2011.
[9] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," ser. PKDD. Springer-Verlag, 2002, pp. 15–26.
[10] T. Kohonen, M. R. Schroeder, and T. S. Huang, Eds., *Self-Organizing Maps*, 3rd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001.
[11] P. Macioek, P. Krl, and J. Kolak, "Probabilistic anomaly detection based on system calls analysis," *Computer Science*, vol. 8, 2007.
[12] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 1998, pp. 604–613.
[13] Nearest neighbors. [Online]. Available: http://scikit-learn.org/stable/modules/neighbors.html
[14] H. Kusumoto and Y. Takefuji, "Self-organizing map algorithm without learning of neighborhood vectors." *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1656–1661, 2006.
[15] S. J. Stolfo, F. Apap, E. Eskin, K. Heller, S. Hershkop, A. Honig, and K. Svore, "A comparative evaluation of two algorithms for windows registry anomaly detection," *J. Comput. Secur.*, vol. 13, no. 4, pp. 659–693, Jul. 2005.