Fault Prediction and Reliability Analysis in a Real Cellular Network

Yash Kumar
International Baccalaureate Candidate
Booker T. Washington High School
Tulsa, USA
4yashkumar@gmail.com

Hasan Farooq, Ali Imran
Department of Electrical and Computer Engineering
University of Oklahoma
Tulsa, OK
hasan.farooq@ou.edu, ali.imran@ou.edu

***Abstract*—Today, the importance of cellular networks is ever-growing. The increasing complexity of networks is expected to decrease reliability. In order to continue reliable operation in a cost-efficient manner, previous literature has explored Proactive Self-Healing methods, but actual application to cellular networks has been lacking. Thus, in this paper, we aim to institute a proactive approach for failure prediction of time series data by surveying a wide range of techniques. To determine the best in predicting network failures, Support Vector Machine (SVM) Regression and multiple Neural Network variants were utilized along with a Continuous Time Markov Chain (CTMC) analytical model to provide reliability analysis. All results are derived from actual network data. We conclude the pattern of these failures is most likely non-linear, and the most promising technique is a Deep Neural Network utilizing Autoencoders. The CTMC analysis demonstrates that current networks barely reside in a healthy state, so the goal is that this paper will lead to improvements, especially in Self-Organizing Networks (SON).**

Keywords—Self-Organizing Networks (SON), Proactive Self-Healing, Fault Prediction, Support Vector Machine (SVM), Neural Networks, Continuous Time Markov Chain (CTMC)

I. INTRODUCTION

In the modern world, the usage of cellular devices in our daily lives is immense. However, the future holds even more growth, and developments could transcend human imagination. This will only bring upon a growth of cellular network activity. While the improvements in our devices continue to increase at a dramatic pace, networks are evolving at a much slower rate. This disparity is resulting in an inefficient system which is not only producing a high number of faults but also causing trouble to the customer. and software that constitute the mobile network. In fact, the faults induce cell outages that ultimately affect the coverage and performance reliability of the cellular network. Faults occur either by base station hardware and/or software malfunctions, power outages, faulty links, multi-vendor incompatibility, or misconfiguration of parameters during network operation. Moreover, the rate of faults is intrinsically proportional to cell density, and complexity of hardware and software that constitute the mobile network. This compromises network reliability, which is defined as the fraction of time the network, including its components, is operational.

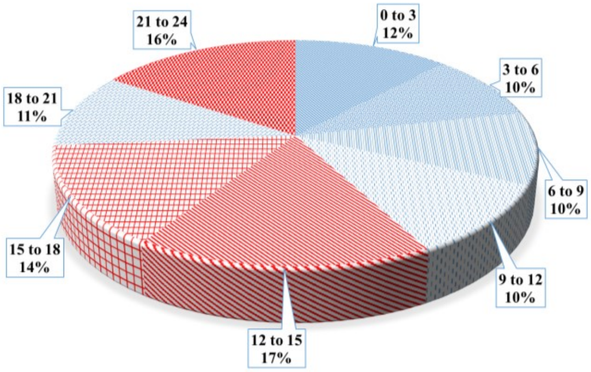


Fig. 1. Percentage of Faults in Given Time Interval

Hence, the results of these faults are a drop in quality of service provided to the customer and a dramatically increased cost for the provider as maintenance times remain high, and the network remains in the sub-optimal state. The problem lies in the current reactive approach utilized by most carriers. This way only responds to the problem and can only hope to limit the magnitude of the problems caused, not prevent it from happening. This method is not experiencing the best success as when the people need to make calls the most, the network experiences the greatest number of faults. From the data set, which was an array of timestamped faults in a month from a national US mobile operator, Fig. 1 was generated.

Fig. 1 displays that during lunch (usually time interval 12 to 15), the greatest number of faults occurred. Also, there was a significant number of faults after lunch (15 to 18). However, there is also a surprise time of 21 to 24 with a high number of faults, and this is believed to occur due to the availability of free minutes during this time which incentivize talking.

Regardless of the time interval, it is important for the customer to have a reliable network when they need to make calls. This is why predictive analysis, a proactive approach, is necessary due to its effectiveness. Intending to deal with the problem before it happens, predictive analysis utilizes mathematical algorithms, such as various modeling techniques, to illustrate the relationship among functioning variables of a system. These models are able to learn the behavior of the system under normal conditions based on these variables, and

978-1-5090-4372-9/17/$31.00 ©2017 IEEE

they can also monitor the patterns that forecast a troublesome scenario. Specifically, in telecommunications, carriers would be able to estimate that a fault is going to occur in a certain amount of time, and thus, could take steps to prevent this fault. This would eliminate any possible difficulties that a customer would experience and is much better than the reactive approach, which can only hope to limit the trouble the customer must suffer.

The increased volume of literature seen for these predictive analysis techniques in recent years is a promising sign. While many papers have looked at different techniques for time series fault prediction [1] [2] [3], this paper has some unique characteristics to differentiate it. The first is the field of telecommunications as many papers have not covered the application of techniques to network failures. Furthermore, this paper applies a wide range of techniques, including both regression models and analytical models, an attribute absent in most papers. Furthermore, most importantly, this paper's most important contribution to current literature is that it utilizes actual data for a network in its calculations. Specifically, the presented analysis is based on one month timestamped faults data from one of the national mobile operators of USA. Many other papers have not had the luxury of having access to real data, which has been a roadblock in analysis.

The goal of this paper is to provide meaningful insight into the success of various techniques for network failure and act as a springboard for future research. Specifically, the application of these techniques to Self-Organizing Networks (SON), defined by [4], could provide a powerful combination which is then able to not only autonomously predict these failures, but also apply self-healing techniques to prevent the problem from occurring. This holds great promise for both the consumer and the provider as it can not only drastically improve the quality of the network, but it can also optimize a network, reducing labor and other costs for network providers.

II. Relevant Works

Existing fault prediction involves the use of several models. However, many of these are only pertinent to specific time-series data and have not utilized cellular network data. [1] utilized several techniques for failure and reliability of time series data and found that Support Vector Machine Regression either outperformed or was comparable to all other techniques tested in the paper. Yet, this paper's data set was limited to car engines. While this paper had a time series data for failure, it cannot be compared to network failures because of the sheer difference in factors which affect failure in engines and factors which affect failure in cellular networks. In fact, telecommunications networks have a much higher number of entities and parameters than engines, resulting in higher fault arrival rates. Furthermore, [2] looked at recurrent neural networks, specifically Infinite Impulse Response Locally Recurrent Neural Networks (IIR-LRNN), and found a novel approach to efficient failure and reliability predictions. However, as previously mentioned, the data set of car engines does not translate to the field of telecommunications. In addition, [3] found that Deep Recurrent Neural Networks, which utilize Autoencoders, perform significantly better by RMSE comparison than other techniques. This paper utilized a time series data; however, it was for energy load forecasting. Moreover, many papers that deal specifically with cellular network reliability have employed analytical models rather than prediction techniques. [5] used event-based reliability model and the Markov process to propose a model for global reliability measurement for telecommunication networks that have link failures, which make them statistically dependent. However, the authors themselves were only able to provide a case study and highlight the need for the use of the actual failure rate and repair rate of a network to gain a better understanding of network failure [5]. Another variation was employed by [6] which proposed a Continuous Time Markov Chain reliability analysis, with a focus on various transient analyses for cellular networks. Continuing with the recurring trend, [6] was only able to demonstrate the model with three case studies due to lack of actual cellular network data. This paper builds on all of these past papers and utilizes elements from each. It combines prediction techniques with analytical models to provide a wide survey of methods and also employs real cellular network data to contribute a better understanding of failure and reliability analysis in the telecommunication field.

III. Methodology

A. Overview

This section investigates different prediction techniques and also utilizes a mathematical technique to formulate a model and various transient analyses. Many of the following techniques were chosen because of past success in previous papers involving time series data [1] [2] [3], but general insight into prediction methods was provided by [7]. The importance of prediction techniques is clear since they are the prerequisite to proactive fault management. Moreover, in order to differentiate among performance by the following techniques, Normalized Root Mean Square Error (NRMSE) was used for the error calculation. This is a frequently used measurement which aggregates individual residuals into a single measure of predictive power. NRMSE provides distinct advantages, the primary being that is a non-dimensional forms of the Root Mean Square Error (RMSE) which increases usefulness by allowing for comparison between RMSE with different units. Furthermore, NRMSEs avoid the highly undesirable use of absolute value in calculations and are able to best reveal differences in model performance due to the higher weight given to unfavorable conditions. While many different methods exist to calculate NRMSE, in this paper, the method used to calculate NRMSE is:

$$NRMSE = \sqrt{\frac{\sum(y_t - \hat{y}_t)^2}{\sum {y_t}^2}} \qquad (1)$$

where y_t is the actual value, and $\hat{y}_t$ is the predicted value.

B. Regression Models

Functional regression models were chosen because they serve as a baseline for comparison with other models for determination of the level of success of the other techniques. Linear regression is one of the simplest methods of prediction as it aims to establish a linear relationship between two variables. In this case, the variables were fault occurrence, represented by n for the n^{th} fault occurrence, and the inter-arrival times in hours, represented by τ_n for the n^{th} fault. The data was divided up into training data, which composed ¾ of the data, and testing data, which composed the remaining ¼ of the data. Using the curve fitting tool in MATLAB, an equation was generated from the training data. This equation was:

$$\tau_n = 0.003454 * n + 2.246 \qquad (2)$$

Exponential Regression is similar to linear regression but instead aims to fit the data to an exponential function with the form $f(x) = a * e^{b*x}$ instead of a linear function. Once again, the curve fitting tool in MATLAB was utilized, and the resulting equation from the training data was:

$$\tau_n = 2.304 * e^{0.001161*n} \qquad (3)$$

C. Equations

Support Vector Machine (SVM) rise from statistical learning theory and are popular learning methods. This technique was chosen because [1] found that SVM Regression was comparable to or outperforming a wide range of techniques on time series data. A SVM model represents points in a space and attempts to map them such that there is a clear space as wide as possible. This space is then divided by a hyperplane for regression purposes. While this hyperplane may be a linear function, SVMs can also use a kernel trick that allows utilization of a non-linear function by mapping inputs into higher-dimensional spaces [8]. For this paper, the SVM Regression was calculated in MATLAB which implements linear epsilon-insensitive SVM (ε-SVM) regression or L1 loss [8]. This includes predictor variables and observed response values in the training data. The SVM attempts to find a function *f*(*x*) that is no greater than an epsilon value, ε, deviation from y_n, the observed response values, for each training point *x*. An optimization problem is formed which is solved in its Lagrange dual formulation in MATLAB [8]. This is obtained by constructing a Lagrangian function from the primal function by introducing nonnegative multipliers α_n and α_n^* for each observation x_n. Then, the Lagrangian function is minimized subject to the constraints:

$$\begin{gathered}\sum_{n=1}^{N}(\alpha_n - \alpha_n^*) = 0 \\ \forall n: 0 \le \alpha_n \le C \\ \forall n: 0 \le \alpha_n^* \le C\end{gathered} \qquad (4)$$

where the constant *C* is the box constraint, a positive numerical value that controls the penalty imposed on observations lying outside the epsilon margin (ε) to prevent regularization. In order to obtain optimal solutions, Karush-Kuhn-Tucker (KKT) complementarity conditions are utilized as the optimization constraints. These conditions indicate that all observations inside the epsilon tube have non-zero Lagrange multipliers (α_n and α_n^*), which lends the name support vectors. Furthermore, the function used to predict new values depends only on the support vector and is given as:

$$f(x) = \sum_{n=1}^{N}(\alpha_n - \alpha_n^*)\, G(x_n, x) + b \qquad (5)$$

where $G(x_n, x)$ is the kernel function. In MATLAB, the kernel function for linear SVM Regression is:

$$G(x_1, x_2) = x_1{}'x_2 \qquad (6)$$

and for Gaussian or RBF Regression, the kernel is [8]:

$$G(x_1, x_2) = \exp(-\|x_1 - x_2\|^2) \qquad (7)$$

D. Neural Networks

Artificial Neural Network (ANN) is a machine learning technique which takes its inspiration from biological neural networks, with both consisting of the same basic components. ANNs are a very popular technique for forecasting, and many variations were utilized to determine which variation had the greatest success. Both [2] and [3] showed high levels of achievement in forecasting ability with their respective techniques, so similar techniques were included. Furthermore, [9] demonstrated the effectiveness of Neural Networks for forecasting in complex systems. ANNs have individual neurons that act as a processing unit by taking in one or more input to product an output. At all neurons, an associated weight is assigned to every input, and this is able to adjust the strength. The neuron then proceeds to sum all the inputs and calculate an output to be passed on. Mathematically, this is represented by [10]:

$$Output = f(i_1w_1 + i_2w_2 + i_3w_3 + \cdots + bias) \qquad (8)$$

For this paper, two applications were utilized in MATLAB. The Neural Network Fitting application and the Neural Network Time Series application were both employed to observe results. Both applications split the training data into a set that consisted of 70% training values, 15% validation values, and 15% testing values. However, the actual performance was computed by using these networks to predict fault inter-arrival times for the testing data, and the resulting values were utilized to compute the error. In the Neural Net Fitting Tool, the number of hidden neurons was experimented with to achieve the best results. Each attempt was replicated three times and then averaged to compute errors. In addition to varying the neurons, two different algorithms, Levenberg-Marquardt and Bayesian Regulation, were utilized. The results presented in the Results section represent only the most successful combination of these two factors. The most success was achieved with the Neural Network having 20 neurons and was trained with Levenberg-Marquardt. The Neural Net Fitting Tool utilizes both fault occurrence and inter-arrival times to construct a network. It utilizes a two-layer feed-forward network. This addition of "hidden" layers between input and output layers requires an activation function, which is usually non-linear, and in this case, is a sigmoid function.

Furthermore, Neural Net Time Series Tool was utilized to create a Nonlinear Autoregressive (NAR) Network. While [2] did utilize IIR-LRNN for its results, it is more suitable to utilize NAR for this data because the desired predictions depend on the past values. NAR predicts fault inter-arrival times given the *d*, the delay value, past values of the fault inter-arrival times. In addition to varying neurons and algorithms, delays were also varied to achieve best results. The most success was achieved with the parameters of 10 neurons, 5 delays, and training with Levenberg-Marquardt. The network remains a two-layer feed-forward network, so the previous math still applies; however, the important difference is that values only depend on fault inter-arrival times and not fault occurrence.

E. Deep Neural Networks

An autoencoder is a type of neural network which consists of an encoder and a decoder. It is trained to replicate its input at its output through these mechanisms. The encoder maps the input to a hidden representation, and the decoder attempts to map this hidden representation back to the original input. This technique was selected because [3] found that autoencoders significantly improved prediction compared to all other neural network techniques. Furthermore, [11] finds that autoencoders tested on probable inputs have small reconstruction error because the autoencoder learns to stay on the manifold by learning salient variations. The autoencoder function was used in MATLAB, and the default 10 neurons in the hidden layer were utilized. The autoencoder was trained on the testing data and then converted into a neural network. This network was then tested on the testing data to compute the NRMSE and RMSE. The training process of an autoencoder is based on optimization of a cost function, which measures the error between input x and its reconstruction at the output $\hat{x}$. The vector $x \in \mathbb{R}^{D_x}$ is mapped to $z \in \mathbb{R}^{D^{(1)}}$ by:

$$z^{(1)} = h^{(1)}(W^{(1)}x + b^{(1)}) \qquad (9)$$

where the superscript (1) is the first layer. $h^{(1)}: \mathbb{R}^{D^{(1)}} \rightarrow \mathbb{R}^{D^{(1)}}$ is a transfer function for the encoder, $W^{(1)} \in \mathbb{R}^{D^{(1)} * D_x}$ is a weight matrix, and $b^{(1)} \in \mathbb{R}^{D^{(1)}}$ is a bias vector. The decoder then maps the encoded representation *z* back to an estimate of the original input vector, *x*, by:

$$\hat{x} = h^{(2)}(W^{(2)}x + b^{(2)}) \qquad (10)$$

where subscript (2) is the second layer. $h^{(2)}: \mathbb{R}^{D_x} \rightarrow \mathbb{R}^{D_x}$ is the transfer function for decoder, $W^{(1)} \in \mathbb{R}^{D_x * D^{(1)}}$ is a weight matrix, and $b^{(2)} \in \mathbb{R}^{D_x}$ is a bias vector. This is the mathematical representation for one layer, but the math remains the same for additional layers of the autoencoder.

F. Continuous Time Markov Chain Analysis

Continuous Time Markov Chain is the analytical model chosen to analyze the data. The model was chosen because [6] proposes various transient analyses that can provide additional insight into the state of the cellular network, which otherwise, would not be available. This is because CTMC can yield more scalable solutions since it is memory-less as only the crux of the past faults is captured by transition probabilities. Furthermore, CTMC can also be used to formulate a predictive model, giving it dual capability. The exploitation of Continuous Time Markov Chain (CTMC) analysis is possible due to the exponential distribution of both the fault inter-arrival times and maintenance times as shown in Fig. 2 and Fig. 3, respectively. The exponential distribution is mathematically tractable, and hence it is possible to use Continuous Time Markov Chain analysis.

For the following analysis, the network can only reside in two states: healthy and sub-optimal. The μ value for fault inter-arrival times is 1/(mean value). In this case, it is 0.38625. The λ value for maintenance time is calculated in a similar way and is 0.01589. Using these values, a generator matrix Q and a rate matrix R can be calculated. The calculations for a two-state system are given as:

$$Q = \begin{bmatrix} -\mu & \mu \\ \lambda & -\lambda \end{bmatrix}$$
$$R = \begin{bmatrix} 0 & \lambda \\ \mu & 0 \end{bmatrix} \qquad (11)$$

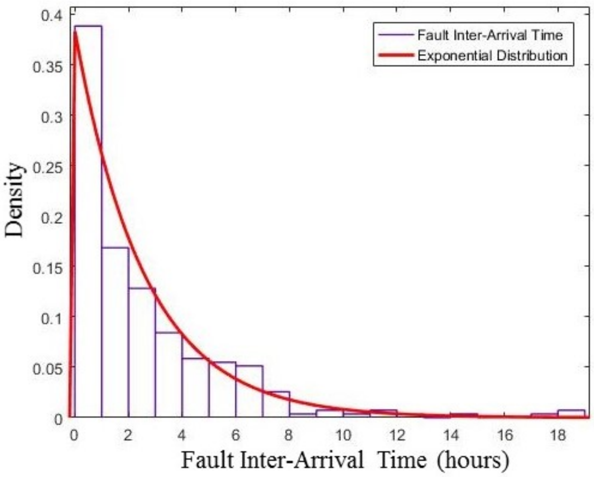


Fig. 2. Exponential Fit of Fault Inter-Arrival Time

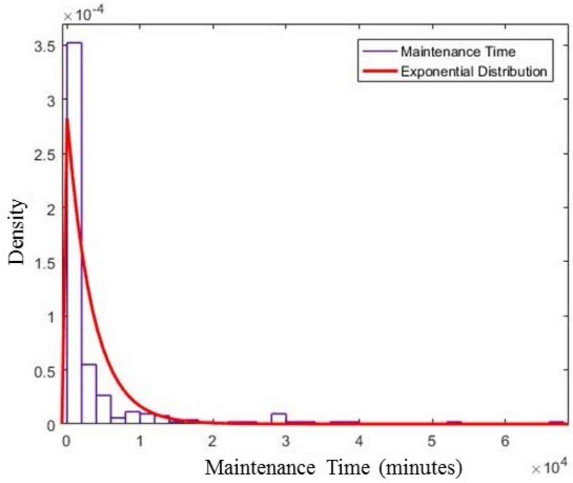


Fig. 3. Exponential Fit of Maintenance Time

Using these two matrices, various transient analyses can be performed. The behavior of CTMC is described by the Kolmogorov differential equation and can be found using the generator matrix Q. Then, the probability vector can be obtained by $P(t) = P(0) * e^{Q(t)}$ with $P(0)$ being the initial probability vector. This paper utilizes the uniformization method because it leads to more efficient computation and results in higher accuracy.

Using this, a probability vector is calculated by:

$$P(t) = \sum_{k=0}^{\infty} e^{-\beta t} \frac{(\beta t)^k}{k!} \hat{P}^k \qquad (12)$$

This series utilizes the Probability Transition Matrix, $\hat{P} = I + Q/\beta$, and the summation can be truncated with the error formula:

$$\varepsilon = 1 - \sum_{k=0}^{M} e^{-\beta t} \frac{(\beta t)^k}{k!} \qquad (13)$$

Algorithms used to truncate the infinite summations to a desired error value were derived from [12]. For this paper, an error value of 0.0001 is utilized. The Probability Transition Matrix forms the foundation for computation of three performance matrices that are utilized in this paper. The first is occupancy time which is computed by placing

$$\Psi_{i,j}(T) = \int_0^T p_{i,j}(t)\,dt \qquad (14)$$

in matrix form, where $p_{i,j}(t)$ is the element of transition probability matrix P.

The second is first passage time, which is the expected time for the system to pass from an optimal state to the suboptimal state. This utilizes the following equation:

$$r_i\zeta_i = 1 + \sum_{j=1}^{N-1} r_{i,j}\zeta_j\,, \qquad 1 \le i \le N-1 \qquad (15)$$

where $i, j \in S$ and

$$r_i = \sum_{j=1}^{N} r_{i,j}\,, \qquad \boldsymbol{R} = [r_{i,j}] \qquad (16)$$

The third is steady-state or limiting distribution. This is defined as $\Psi = [\psi_1 \quad \psi_2]$ where

$$\Psi_j = \lim_{t\to\infty} \Pr(X(\mathrm{t}) = j) \qquad (17)$$

and

$$\psi_j r_j = \sum_{i=1}^{N} \psi_i r_{i,j}$$
$$\sum_{i=1}^{N} \psi_i = 1 \qquad (18)$$

Solving this yields the steady state distribution for the network.

Furthermore, a model was generated from the transient probability vector. The model was developed by calculating the probability of a fault after a certain amount of time and considering it a detection of a fault if the value met a designated threshold level. For this paper, the model was checked every 4 hours with a threshold level of 75%. The model was awarded a 1 for correct predictions, 0 for incorrect predictions, and a resulting percentage was calculated. This model can also be updated by adjusting values with new prediction values, with techniques such as exponential smoothing. However, this is irrelevant for this model because the probability transition matrix values are too high to be effected since maintenance time is much higher than fault inter-arrival time.

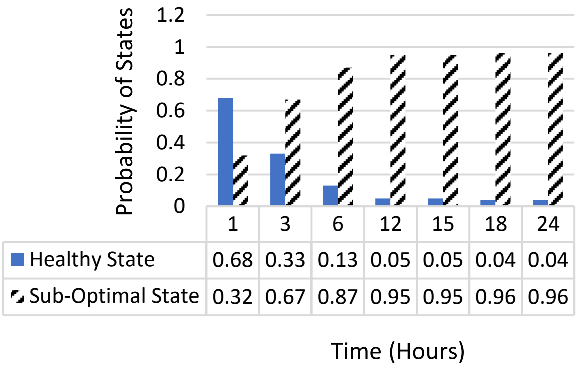
TABLE 1 RESULTS OF PREDICTION TECHNIQUES

Results	**Performance on Test Data**		
	Technique	***NRMSE***	***RMSE***
1	Deep Neural Network with Autoencoders	0.122092	0.504425
2	Nonlinear Autoregressive Neural Network	0.776133	3.206542
3	Gaussian Kernel Support Vector Machine Regression	0.783231	3.235869
4	Exponential Regression	0.783803	3.238231
5	Linear Regression	0.784666	3.24798
6	Neural Network	0.792844	3.29789
7	Linear Support Vector Machine Regression	0.802023	3.313506

IV. RESULTS

The results from the models are summarized in Table 1. The results indicate that most of the models had relatively the same success in predicting the fault's inter-arrival time on the test data, besides the deep neural network with autoencoders. The huge improvement in NRMSE demonstrates significantly better effective fault prediction than all other techniques. This is explained by [13] as it discusses that pretraining each layer with an unsupervised learning algorithm can allow for better initial weights, and thus is why autoencoders produce drastically better results. In addition, the other significant conclusion is that the linear models performed worse than their nonlinear counterparts. This shows that the data does not follow a linear trend and is much more complicated. Furthermore, while most of the NRMSE values are relatively high, it should be noted that since the models attempt to predict the failures of the whole network instead of one base station, it is expected.

Furthermore, the results of the CTMC analysis provide crucial information about the reliability of the entire network. These analyses can be adapted each time for new values in order to compute new expected times for occupancy times, the first occurrence of a fault, and steady-state distribution. The transient analysis using the probability matrix in Fig. 4 shows that the probability that the network switches from healthy to unhealthy is very high. In fact, after 12 hours, there is a 95% chance that the system is in an unhealthy state.



	1	3	6	12	15	18	24
■ Healthy State	0.68	0.33	0.13	0.05	0.05	0.04	0.04
▨ Sub-Optimal State	0.32	0.67	0.87	0.95	0.95	0.96	0.96

Fig. 4. Transient Analysis for First Day of Network

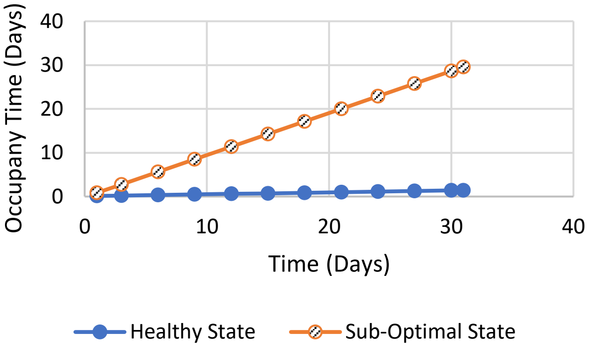


Fig. 5. Occupancy Times for a Month

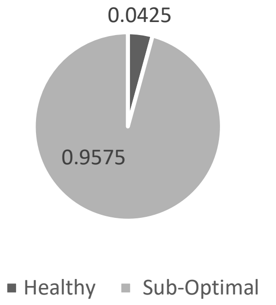


Fig. 6. Steady-State Distribution for Lifetime of Network

After one day, the value for sub-optimal state remains constant at 0.9605. However, using these probability values, a possible model can be made. For demonstration, a model was developed that used a 75% threshold level to signify that a fault has occurred. The model was checked every four hours for a week, and the prediction accuracy was given a 1 for a correct prediction and a 0 for an incorrect prediction. The model's accuracy was a 27/42 or 64.29%. Another advantage provided by this model is that it avoids type 2 errors which cause the most amount of damage.

The results of the occupancy time analysis presented in Fig. 5 show that the network for a month's time (31 days) will spend on average only 1.3 days in the healthy state and 29.7 days in a sub-optimal state. This shows the need for networks to have a better proactive approach in order to reduce the number of faults occurring in order to improve quality for the customer. The first passage time was calculated to be 2.589 hours which is in accordance with the mean fault inter-arrival time of 2.589 hours. This makes theoretical sense as the CMTC analysis is based on the fact that both variables follow an exponential distribution.

The steady-state distribution as illustrated in Fig. 6 also confirms the need for a proactive approach. The distribution found that during its lifetime, the network will spend only 3.95% of it time in the healthy state, while it will spend a massive 96.05% of its time in a sub-optimal state. Such a high sojourn time in the sub-optimal state is due to the fact that we considered fault series data from multiple base stations instead of a single base station. This highlights the fact that massive densification, aimed for 5G, is consequently going to increase the fault arrivals and Proactive Self-Healing, capable of forecasting network faults in advance before subscribers are affected, is very much needed for reliable operation in 5G cellular networks.

V. Conclusions & Future Work

The results provided by the real data highlight some troubling phenomena. They confirm the ineffectiveness of the current reactive techniques as the transient analysis demonstrate the very high probability of transitioning from healthy to sub-optimal state and a lifetime expectancy of only being in the healthy state 4% of the time. This is bad news for both the consumer and the network provider, and it underscores the need for a change in approach. Predictive techniques might be that answer. While most did not have great success, it is clear that nonlinear models did better than linear counterparts. This shows that the pattern is complex and non-linear.

Thus, it is logical that the most successful technique was Deep Neural Network with Autoencoders, and hence shows the most promise for future application in SON. Moreover, the scope of the current work is limited to evaluating prediction accuracy, and the data was considered cumulatively for a cellular network. However, for future work, the same method can be applied for a per-base station study. The per-base station data can be used to localize faults so that once the predicted fault occurrence time is near, the network prioritizes the verification of each base station or initiates the configuration of parameters. This can be further complemented by proactive cell outage compensation algorithms detailed by [14], which by knowing future fault arrival times, can take preemptive actions to reconfigure the network. This will prevent outages from occurring and help the network continue to run reliably. Future research regarding the application of other deep neural network learning techniques should also be explored as a possibility.

VI. Acknowledgements

This work was made possible by NSF Grant No. 1619346. The statements made herein are solely the responsibility of the authors.

References

[1] M. Moura, E. Zio, I. Lins, and E. Droguett, “Failure and reliability prediction by support vector machines regression of time series data,” Reliability Engineering & System Safety, vol. 96, no. 11, pp. 1527-1534, 2011.

[2] E. Zio, M. Broggic, L. R. Goleab, and N. Pedronib, “Failure and Reliability Predictions by Infinite Impulse Response Locally Recurrent Neural Networks,” Chemical Engineering Transactions, vol. 26, pp. 117-122, 2012.

[3] E. Busseti, I. Osband, and S. Wong, Deep Learning for Time Series Modeling, 1st ed, 2012.

[4] O. G. Aliu, A. Imran, M. A. Imran and B. Evans, “A Survey of Self Organisation in Future Cellular Networks,” in IEEE Communications Surveys & Tutorials, vol. 15, no. 1, pp. 336-361, First Quarter 2013.

[5] W. Chen, N. Huang and R. Kang, “A reliability model with the dependent failures for telecommunication network,” 2009 8th International Conference on Reliability, Maintainability and Safety, pp. 1129-1132, Chengdu, 2009.

[6] H. Farooq, M. S. Parwez and A. Imran, “Continuous Time Markov Chain Based Reliability Analysis for Future Cellular Networks,” 2015 IEEE Global Communications Conference (GLOBECOM), pp. 1-6, San Diego, CA, 2015.

[7] F. Salfner, M. Lenk, and M. Malek, “A survey of online failure prediction methods,” CSUR, vol. 42, no. 3, pp. 1-42, 2010.

[8] “Understanding Support Vector Machine Regression - MATLAB & Simulink,” Mathworks.com, 2016.

[9] W. Wu, F. Zhang, M. Liu and W. Shen, “A multi-agent based failure prediction method using neural network algorithm,” 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2268-2272, San Diego, CA, 2014.

[10] T. Mitchell, “Machine Learning,” McGraw Hill, 1997.

[11] R. Socher and C. Manning, “Deep Learning for NLP (without Magic),” NAACL Atlanta, 2013.

[12] V. Kulkarni, Introduction to modeling and analysis of stochastic systems. New York: Springer Science+Business Media, LLC, 2011.

[13] Q. Le, A Tutorial on Deep Learning Part 2: Autoencoders, Convolutional Neural Networks, and Recurrent Neural Networks, 2015.

[14] O. Onireti, A. Zoha, J. Moysen, A. Imran, L. Giupponi, M. Ali Imran and A. Abu-Dayya., "A Cell Outage Management Framework for Dense Heterogeneous Networks," in IEEE Transactions on Vehicular Technology, vol. 65, no. 4, pp. 2097-2113, April 2016.