Access Network Data Analytics

(Machine Learning Applied to Cable Access Data)

A Technical Paper prepared for SCTE/ISBE by

Karthik Sundaresan
Principal Architect
CableLabs
858 Coal Creek Circle
Louisville, CO 80027
303-661-3895
k.sundaresan@cablelabs.com

Jay Zhu
Engineer
CableLabs
858 Coal Creek Circle, Louisville, CO, 80027
303-661-3312
j.zhu@cablelabs.com

Table of Contents

List of Figures

List of Tables

Introduction

The cable access network has large amounts of monitoring data collected across various network equipment (CMTS, CMs, STBs, APs etc.). This data reflects the state of the network, status of devices and outside plant. Some examples are PNM data (RxMER, Channel coefficients), CM and CMTS MIBs (FEC stats, Service flow stats, packet drop counts), IPDR data, etc. Now, given this data set, how can the operator identify network plant issues, reliably, proactively and automatically?

Combining the analysis of data along, with network topology and device location, it is possible to create a general view of the plant condition and isolate problem sources. This paper tackles questions around, how can the operator identify network plant issues on a single modem, across a serving group, across fiber node, and correlate data across all devices in a reliable, proactive and automatic way. Machine-learning / data-analysis applications can crunch through layers of data to identify patterns and prioritize network issues automatically, allowing operators to reduce troubleshooting and problem resolution time, thereby reducing operational costs and enhance network reliability.

Problem Statement

1. Lots of data, no knowledge

There are various areas in the cable network space, each with large sets of data, to which various Machine Learning techniques can be used to solve various problems.

Proactive Network Maintenance (PNM) involves processing vast amounts of fine-grained information about the state of the network to look for patterns that indicate impending problems. Current PNM solutions have focused on leveraging the physical-layer channel measurements performed by DOCSIS cable modems and CMTSs, along with knowledge of the HFC plant topology and well-known digital signal processing algorithms in order to locate physical degradations of the plant. ML algorithms open up this space to identify a much wider variety of network impairments, particularly when those impairments aren’t predictable using a priori knowledge.

CMTS and cable modem features and capabilities can be leveraged to enable measurement and reporting of network conditions so that undesired impacts like plant equipment and cable faults, interference from other systems, and ingress, can be detected and measured. With this information, cable network operations personnel can make modifications necessary to improve conditions and monitor network trends to detect when network improvements are needed.

Some of the PNM measurements are very valuable in that they can reveal problems in the network. We illustrate a couple of examples to which we can effectively apply Machine Learning techniques. This will help in quickly and automatically identifying problems, instead of a human operator looking through the data and flagging issues manually.

2. Why Machine Learning

A CM with full-band tuner capability acts like a spectrum analyzer. The spectrum analysis reveals suck-outs, excessive tilt, frequency peaking, unwanted filters, FM (or other) radio ingress, early roll-off, etc. Each of these issues (signal patterns) can be learnt by a Machine Learning application, which can then monitor live signals to flag problems in the network.

Machine Learning algorithms can look for abnormalities from the data obtained from each CM and across all CMs in the network. Grouping of CMs with common problems can be used to identify issues affecting multiple subscribers and isolate the cause of an issue in the cable plant.

The data sources which we start off with is primarily is the PNM data available from the cable modem today. We can add to that DOCSIS MAC layer and packet statistics along with CM status objects (e.g. T3/T4 timeouts, partial service indicators etc). Understanding the channel response characteristics can also help with creating optimal D3.1 profiles. All this can be combined to define a "health metric", a human readable shorthand to identify the status of a CM. Visualization of data is a powerful first step in understanding data and this paper will also discuss viable options to visualize data for the operators. This paper will discuss methods for anomaly detection, root cause analysis, historical behavior analysis, pattern recognition, classification, prediction and condition evaluation in the access network data. See paper refernces [1] & [2] for further details on how machine learning applies to access network data analytics

Data sources

3. Sources (how)

The DOCSIS 3.0 and 3.1 technologies introduce CMTS and cable modem features and capabilities that can be leveraged to enable measurement and reporting of network conditions. This helps to detect and predict undesired impacts such as plant equipment and cable faults, interference from other systems and ingress. The goal is to rapidly and accurately characterize, maintain and troubleshoot the upstream and downstream cable plant, to guarantee the highest throughput and reliability of service. See [7](DOCSIS PHYv3.1).

3.1. Data types (what)

There are a few powerful indicators for problems in a cable network. These include details such as excessive transmit power at the CM or low receive CM signal levels. There were also indicators for high packet or codeword errors, both at the CM and CMTS, and low receiver MER (modulation error rate, a measure of noise and interference in the signal. While these indicators are useful, they are a view of a single endpoint in the network and don't tell an operator exactly what is the problem. It will be useful to look at these indicators not only from the point of view of a single end-device, but holistically across the entire network to understand the root causes.

Some of the data types which are useful for such analysis are

- CM,CMTS RxMER level per subcarrier
- CM, CMTS Transmit & Receive power levels

- FEC Statistics (Correctable vs Uncorrectable)
- Channel pre-equalizer Coefficients
- Downstream vs Upstream parameters
- Network Topology
- Topology map, Latitude/longitude of the devices
- Overlay map of local Wireless channels and their frequencies (e.g. LTE channels)

3.2. PNM Data (TFTP)

CMs and CMTS can upload Captured PNM Data to the configured PNM Server via TFTP. These PNM data can include the following measurements. There are MIB objects to configure/trigger the file transfer and measurements.

DOCSIS Downstream PNM Measurements and Data include: Symbol Capture, Wideband Spectrum Analysis, Noise Power Ratio (NPR) Measurement, Channel Estimate Coefficients, Constellation Display, Receive Modulation Error Ratio (RxMER) Per Subcarrier, FEC Statistics, Histogram, and Received Power.

DOCSIS Upstream PNM Measurements and Data include: Capture for Active and Quiet Probe, Triggered Spectrum Analysis, Impulse Noise Statistics, Equalizer Coefficients, FEC Statistics, Histogram, Channel Power, and Receive Modulation Error Ratio (RxMER) Per Subcarrier.

3.3. SNMP data

The CMs and CMTSs also reflect a lot of the PNM Data measurements to via MIB objects. These include the following measurements/data objects:

- CM Downstream Objects
  - CmDsOfdmSymbolCapture
  - CmDsOfdmChanEstimate
  - CmDsOfdmMerForCandidateProfile
  - CmDsOfdmRequiredQamMer
  - CmDsOfdmHistogram
  - CmDsOfdmRxMer
  - CmDsOfdmFecSummary
  - CmDsOfdmRequiredQamMer,
  - CmDsOfdmMerMargin,
  - CmSpectrumAnalysisMeas
  - CmDsConstDispMeas,
  - CmUsPreEq,
  - D3.0 full band capture
  - CmDsHist
- CM Upstream Objects
  - CmUsOfdmaEqualizerCoefficients
- CMTS Downstream Objects
  - CmtsDsOfdmSymbolCapture
  - CmtsDsOfdmNoisePowerRatio
- CMTS Upstream Objects
  - CmtsUsOfdmaActiveAndQuietProbe
  - CmtsUsOfdmaImpulseNoise
  - CmtsUsOfdmaHistogram
  - CmtsUsOfdmaRxMerPerSubcarrier
  - CmtsUsSpectrumAnalysis

Solution Approach

4. Understand the problem (Manual)

4.1. Data Visualizations

Data visualization describes the effort to help understand the significance of data by placing it in a visual context. Patterns, trends and correlations that might go undetected in text-based data can be exposed and recognized easier by data visualization.

A picture is worth a thousand words, only when the story is best told graphically rather than verbally and the picture is well designed. One could stare at a table of numbers all day and never see what would be immediately obvious when looking at a good picture of those same numbers.

A table of CM RxMER values can do two things extremely well: it expresses the MER values precisely and it provides an efficient means to look up values for a particular subcarrier and time. But if we're looking for patterns, trends, or exceptions among these values, if we want a quick sense of the plant state contained in these numbers, or we need to compare whole sets of numbers rather than just two at a time, a table approach fails.

Instead of studying single snapshots of topical issues or events in ever-greater detail, however, we can create “datascopes” that can be used to zoom in and out of large data sets in search of new understanding.

In this paper, we are primarily working with CM RxMER values obtained from a live field trial DOCSIS 3.1 plant. The DOCSIS 3.1 OFDM Channel here is 96 MHz wide, i.e. 1920 (50Khz) subcarriers. The number of Active sub carriers within the channel is ~1750 subcarriers. Due to errors in the data capture, or misalignment in the channel set up across different CMTSs, we chose to manually best align the MER values across the different CMs. (set of Data subcarriers were shortened to less than 1750.)

The primary goal of the data visualizations here is to understand the current Channel and plant conditions and observe patterns visually for a CM over a time period and for sets of modems at a given time and for sets of modems across time. The idea is to see if there are common issues to identify (and the answer is yes.)

The idea is to take the raw data and create visualizations that help understand the plant state. We used a mix of 2d and 3d plots, to understand patterns over frequencies, over time and over topology. We are learning from the data about the kinds of issues present and how best we can automatically detect them.

We are using mainly RxMER data, as that was the data available to us, but some of these principles (visualizations and automatic anomaly detection) will apply to other data types as well.

4.2. CM RxMER Visualizations

The below Figures show the CM RxMER Data plots. The x-axis is the sub-carrier frequency, the y-axis is the time over which the captures were done, and the vertical z-axis is the actual reported RxMER db values per sub-carrier.

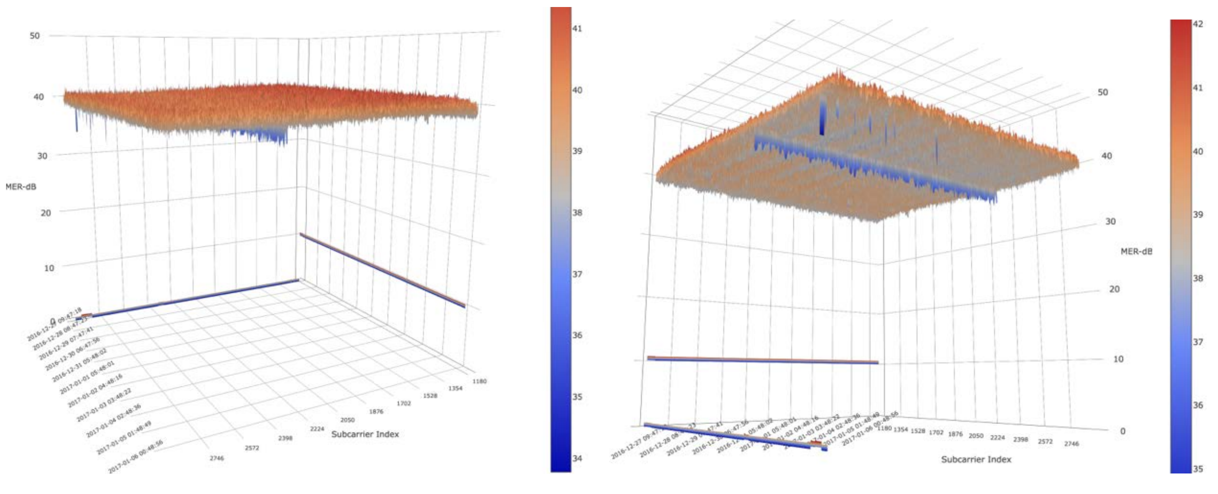

Figure 1 – CM RxMER data over frequencies and time (Same CM)

Figure 1 shows the CM's RxMER values which looked to be centered around 40dB, most of the values are near that, though one can see a small dip in the RxMER values around subcarrier index 2398-2224

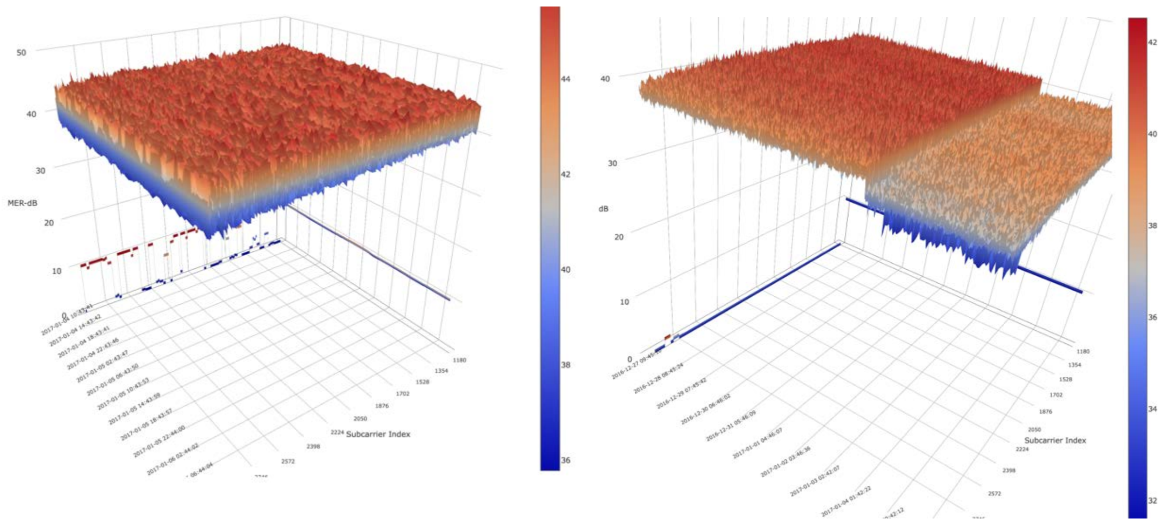

Figure 2 – CM RxMER (2 different CMs, variations and sudden drop)

Figure 2 shows the two different CM's RxMER values, the one on the left look to be centered around 41dB, but show a lot more variability in frequency and in time (max of 45db to min of 37 db. The second modem shows RxMER values which are centered at 40db for a few weeks, and then jumps down to ~37db going forward.   This essentially means some event happened in the network causing a drop in the RxMER. (Was this the effect of perhaps a splitter added ahead of the CM in the home network, or could it be something else ?)

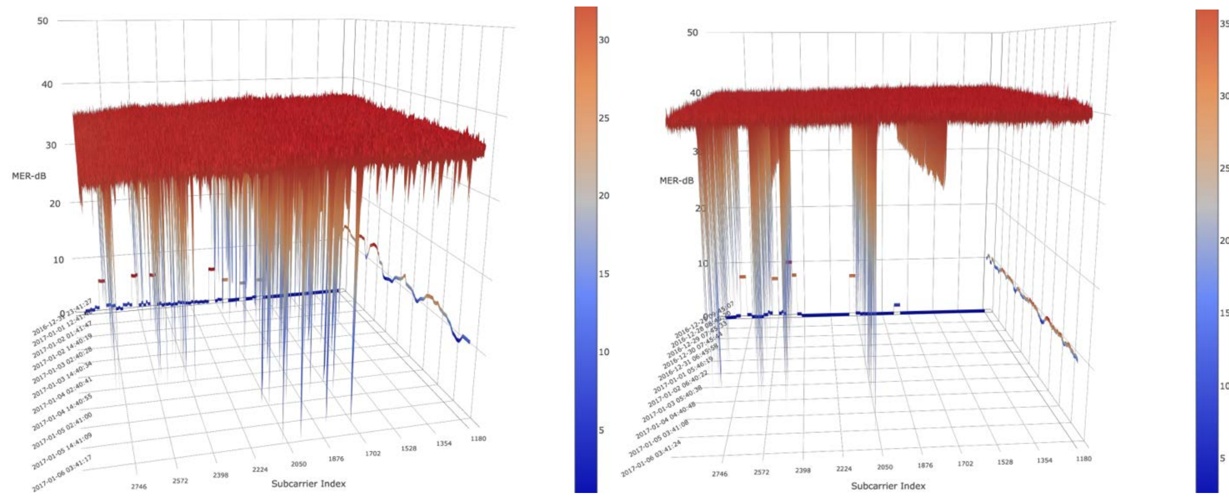

Figure 3 – CM RxMER data , differnet examples of issues

Figure 3 shows the two more CM's RxMER values, with each CM showing issues at specific sets of subcarriers, and in some cases over time.

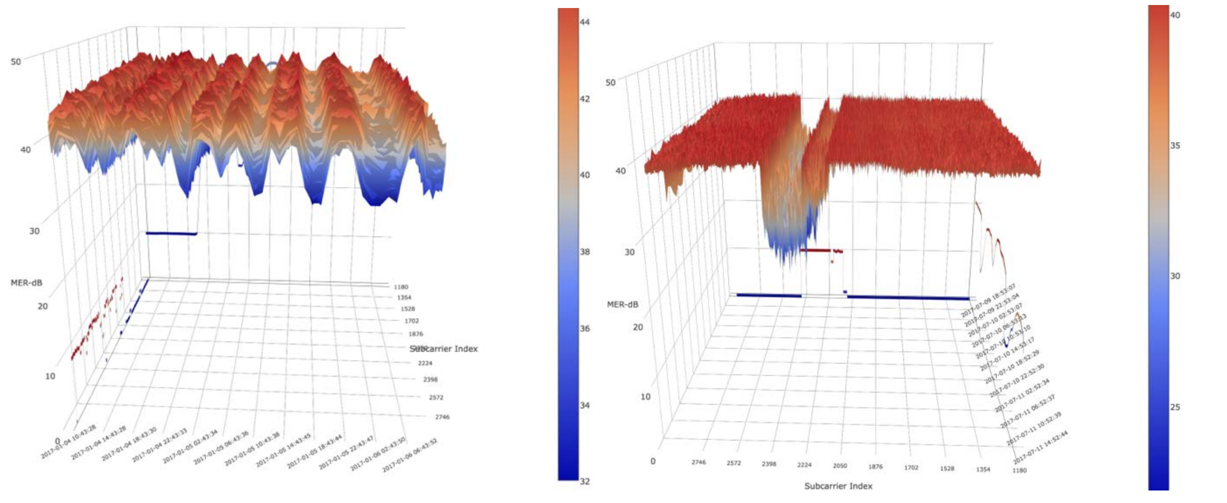

Figure 4 – CM RxMER data (oscillations over time, interference)

Figure 4 shows the two more CM's RxMER values, with the left CM showing issues in a somewhat periodic fashion, with the average RxMER value oscillating up and down over time. The second CM is showing issues over a certain set of frequencies, in this case this happens to be about 300 subcarriers or 15 MHz wide. Now as example of the possibilities of automatic issue identification: Some of the wider LTE channels are either 5,10,15 or 20 MHz wide, so if one could co-relate this interference issue seen on a set of CMs at a specific frequency to the local LTE frequency/channel map corresponding to the CM's physical locations, one might be able to classify this issue as LTE interference and then go look for ways to mitigate it.

Hopefully the above sub-set of CM RxMER plots gives the reader a bit of understanding into the plant and how the plant response varies over frequencies and time.

4.3. Map based visualizations

Another type of visualization is geographical-location based. Here the idea is to overlay the per-CM PNM data on a map, where plant layout, fiber node locations and the modem locations are known and mapped. In the figures shown below the CMs are grouped and analyzed hierarchically under the fiber node serving that location.

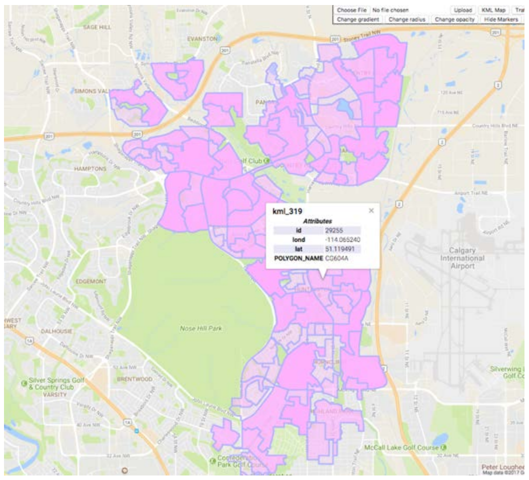
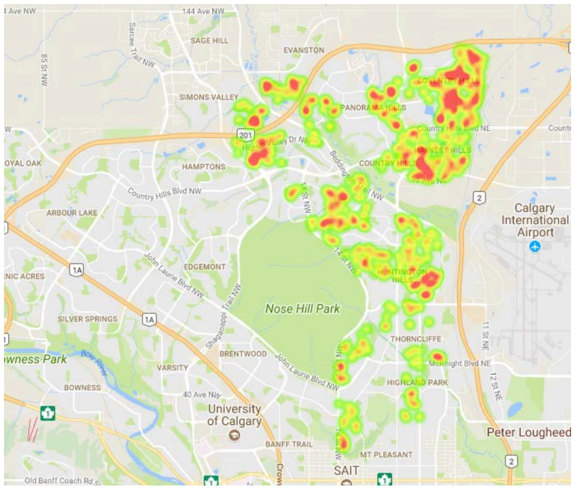

Figure 5 – CMTS Serving FiberNode locations, Heat Map of CM Rx Power level

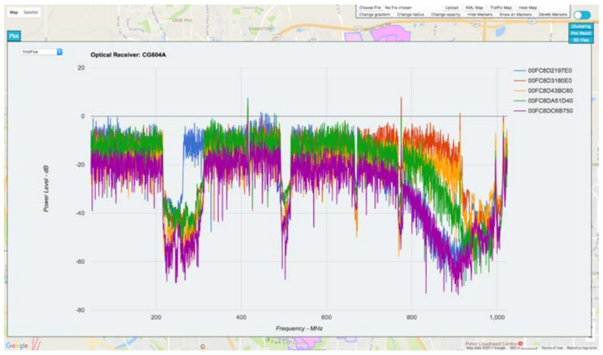
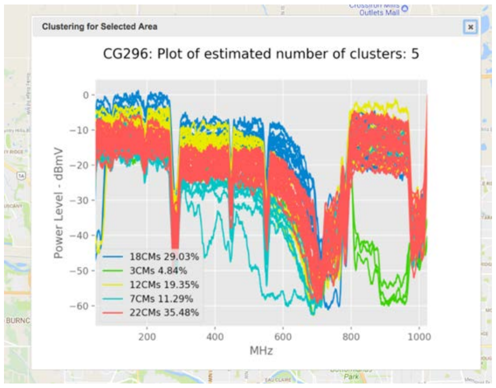

Figure 6 – CM Rx Power level over 1GHz spectrum, & K-Means clustering of all CMs based on RxPower level

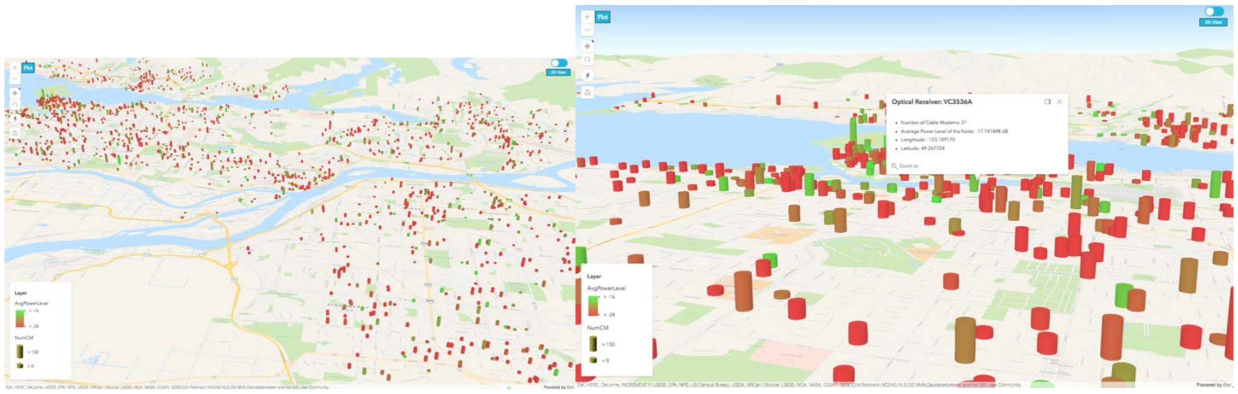
Optical Receiver: VC3536A
Number of Cable Modems: 21
Average Power Level of the Node: -17.741498 dB
Longitude: -123.109170
Latitude: 49.267724
Zoom to
Layer
AvgPowerLevel
NumCM
3D View

Figure 7 – Map of Fiber nodes with CM Rx Power level grouped together

5. Machine learning approch to Anomaly detection

5.1. Preparing the data

Machine learning algorithms learn from data. It is critical to feed the learning algorithms the right data for the problem one wants to solve. Even if there is good data, one needs to make sure that it is in a useful scale, format and meaningful features are included.

The process for getting data ready for a machine learning algorithm can be summarized as: Select Data, Preprocess Data, Transform Data.

The selection step is concerned with selecting the subset of all available data that you will be working on. One needs to consider what data is actually needed to address the problem.

Three common data preprocessing steps are formatting (data may be in a proprietary file format and you would like it in a relational database or a text file), cleaning (removal or fixing of missing data.) and sampling (a smaller representative sample of the selected data).

Three common data transformations are scaling, attribute decompositions and attribute aggregations. This step is also referred to as feature engineering.

Data preparation can involve a lot of iterations, exploration and analysis.

5.2. Machine Learning approach

The machine learning approach to automatic anomaly detection we have taken in this paper is as follows:

Using combined history of all CM's RxMER data, we identify the different anomalies in the CM's RxMER data, and use that to train a Machine learning classifier. We use this classifier to quickly predict and the label issues on new incoming data. We then analyze the patterns in these anomalies to detect system wide patterns and generate operational information/knowledge for the operator.

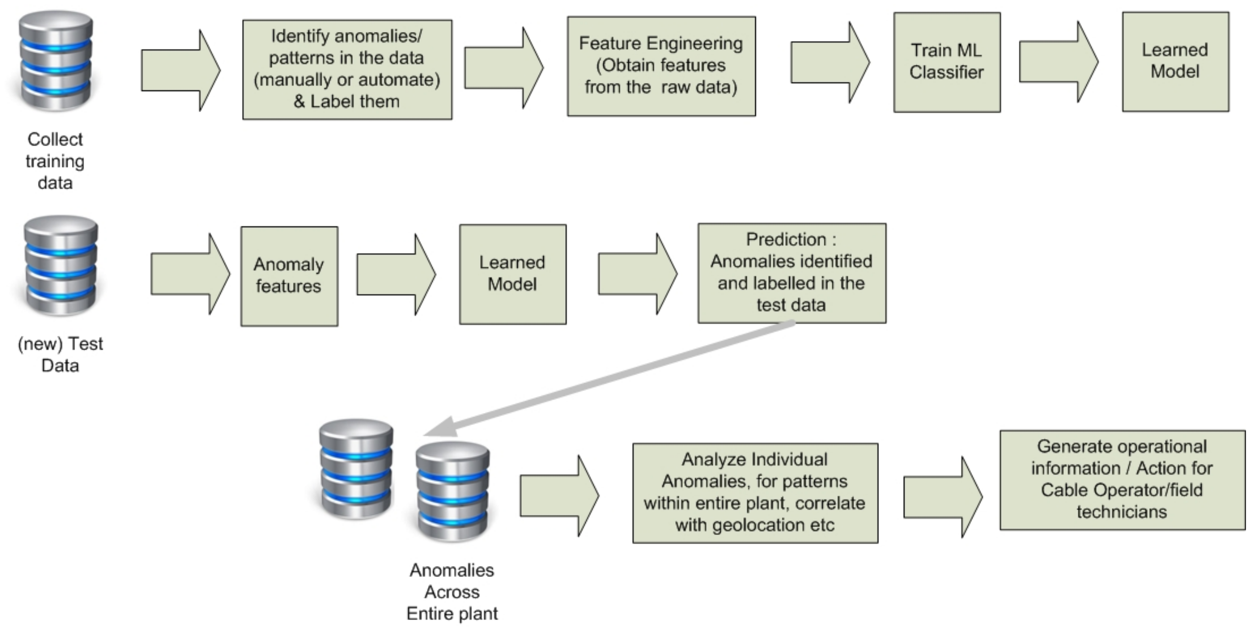
Collect training data
Identify anomalies/ patterns in the data (manually or automate) & Label them
Feature Engineering (Obtain features from the raw data)
Train ML Classifier
Learned Model
(new) Test Data
Anomaly features
Learned Model
Prediction : Anomalies identified and labelled in the test data
Anomalies Across Entire plant
Analyze Individual Anomalies, for patterns within entire plant, correlate with geolocation etc
Generate operational information / Action for Cable Operator/field technicians

Figure 8 – Machine learning approach

We divided the data samples we had into training (70%) and test (30%) sets.   To train a machine learning classifier we need events identified and labeled for the classifier to train on.

One could do this manually but this would be very kind and labor-intensive asset. One would need to walk through thousands of samples from each cable modem to identify issues and then label them. Manual labeling is too slow and doesn't allow for experimental changes as one goes through the process.

We decided to implement an anomaly detector which will automatically identify issues in the CM's RxMER data sample and label them into somewhat generic buckets of issues. In this paper we chose tilt, roll-offs, Sharp MER drops, and wide MER drops as a starting point in terms of the set of issues we wanted to identify. In the future, this set of labels or issues can be extended two specific patterns which we want to identify in the RxMER data.

Once the anomaly detector detects and labels patterns, we reduce the data by extracting the features of the raw data. In this case, we are extracting features from the anomaly such as the width, the depth, the average drop in RxMER, etc. to create a subset of features which the ML classifier will learn on instead of all the raw data.

5.3. Initial Anomaly Detection: Sliding Median & threshold comparison

A moving average is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. The threshold between short-term and long-term depends on the application, and the parameters of the moving average will be set accordingly. Viewed simplistically it can be regarded as smoothing the data. From a statistical point of view, the moving average, when used to estimate the underlying trend in a time series, is susceptible to rare events such as rapid shocks or other anomalies. A more robust estimate of the trend is the simple moving median over n time points.

A moving median is less sensitive to outliers, where an outlier is usually a single point in time series that is very different from all others, which may be due to some kind of error. Moving median filter simply removes outliers from the result, where moving mean/average always takes into account every point. However, moving median can be even more sensitive to short-term significant spikes that span several points, especially when they span more than half of the moving window.

The current implementation for anomaly detection on a CM's RxMER values is as follows:

For a single CM RxMER sample, maintain a sliding window with a set of different window sizes. The window slides through all subcarriers on both directions from a low frequency to high frequency and from high frequency to low frequency. It generates a list of median MER values of each window position, which is smoother than the original sample and can be used for shape comparison to extract anomaly events (dips) from the original MER sample very precisely. With different window sizes, the sliding median algorithm can look at the sample from different scales to cover as much anomaly events as possible. In the current implementations, we use 2 window sizes of 200, 800 subcarriers. The other option is to use a Savagol filter (see reference [6], for a description of Savitzky-Golay Filtering) which removes very high frequency noise from the data.

In this paper we chose rhe sliding median algorithm , it is used to detect the anomalies, essentially any deviations below the sliding median RxMER value when comparing values from the original sample. The algorithm flags any subcarrier values below with threshold and these are marked as anomalies.

5.4. Feature Extraction

If there are many independent features that each correlate well with the class, machine learning is easy. On the other hand, if the class is a very complex function of the features, one may not be able to learn it. Often, the raw data is not in a form that is amenable to learning, but one can construct features from it that are useful. It is advantageous to reduce the number of features considered to focus on a subset of particular interest; this is called feature selection. After anomaly to detection, for each anomaly, we reduce each sample to a set of features which define that sample. This process is feature engineering or dimensionality reduction. Using the best and the least features to describe a learning problem is very important in Machine Learning, this is primarily for improving accuracy of the analysis. It also reduces measurement costs, creates faster systems with less memory, and allows simpler interpretation of the results.

Here we take a few relevant features which describe the anomaly accurately instead of the whole raw anomaly data sample. In particular, the features we use are the area occupied by the anomaly, the width (in subcarriers), lowest MER value, the average MER value at the bottom of the anomaly, the start MER level & end MER levels of the anomaly etc. In order to get rid of noise and to let the classifier be focused on certain aspects from the anomaly events, once the anomaly detector finds and extracts anomalies from the RxMER sample it immediately extracts features / regions of interest by analyzing the event width, starting MER value and ending MER value (relative), MER level of the bottom of the event, and the event score (the accumulative area / db below to the threshold) etc.

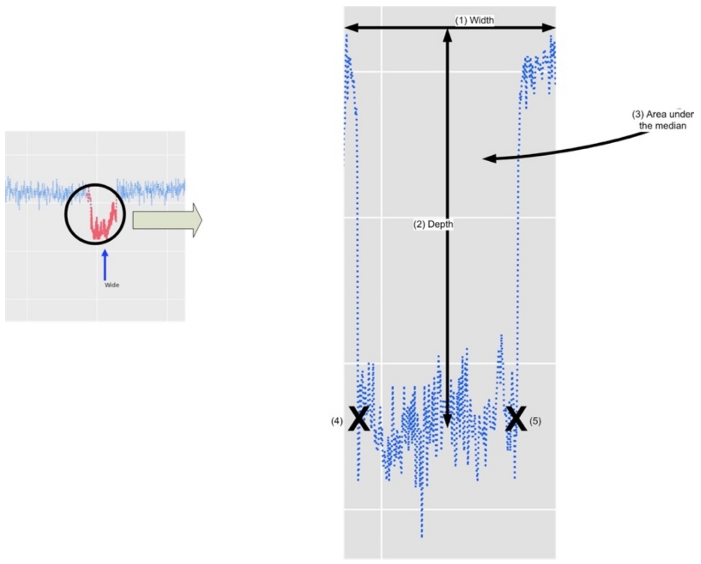

Figure 9 – Feature engineering approach

These extracted features from each event can be used as the input (after labeling) to train and test an intelligent classifier.

5.5. Data labeling

Unlabeled data consists of samples of natural or human-created artifacts that you can obtain relatively easily from the world. Labeled data typically takes a set of unlabeled data and augments each piece of that unlabeled data with some sort of meaningful "tag," "label," or "class" that is somehow informative or desirable to know. ML problems start with data — preferably, lots of data (examples or observations) for which you already know the target answer. Data for which you already know the target answer is called labeled data. In supervised ML, the algorithm teaches itself to learn from the labeled examples that we provide.

The labeling approach we take here can be based on clustering of the basis of the raw samples or clustering based on extracted features. Clustering based on the extracted features, was faster and more reliable. We use a simple K, means clustering to manually label the groups and combine them into larger ones. With the fact that similar patterns exist in the anomalies we detected, we can cluster the anomalies into a few major groups. The noise in the anomalies can be an issue when trying to use K-means to automatically generate clusters with a small value of K. Increasing the value of K can prevent considerable impact from the noise since the K-means will create much smaller groups, and therefore, makes much less clustering mistakes and reduces the number of samples, which would help generate much cleaner visualizations for manual identification and labeling.

For the smaller clusters generated by K-means, we manually check each of them and group similar ones into larger clusters. For those smaller clusters who have too much noisy samples and mis-clustering samples, we remove them to keep the dataset clean. We then give each large cluster a numeric label and they are ready to be used as the input for training classifiers.

For now, we are limiting the work to 4 simple labels, the idea in the future is to increase the label space. Temporarily, we have 4 labels for all the anomalies samples. However, the anomalies should be classified more clearly in the future. For example, from our observations, the wide anomalies could be separated into a few smaller groups because some of them have slightly different patterns and they could be caused by different interference sources.

Based on thresholds for each of the anomaly types, we differentiate each incoming sample as tilt (up vs down), roll-offs (left end or right end), sharp MER drops, and wide MER drops). Each sample from the anomaly detector is labeled as one of these anomalies. The thresholds are set such that some of the samples are discarded, so that with the labelled samples we use to train Machine learning classifiers, we use very clean data to learn on. A manual review of these labels is performed as a sanity check.

The labeling operates in stages, each stage identifying the labels for that type of anomaly.

5.5.1. Synthetic data generation

Synthetic data is an alternative when you don’t have enough actual data samples for training. Once a general pattern has identified in the actual data, which is distinctive and can be described, one can create pseudo-random synthetic data samples which follow the same pattern characteristics. This synthetic data is useful to train a machine learning classifier to learn the properties of known features, for future identification.

By learning from the data, it’s possible to abstract the similarity of patterns and guess what they should look like with random changes. By doing so, we developed out synthetic RxMER data generator that can randomly create anomaly events (sharp anomalies, wide anomalies, tilts, roll-offs) and put them into RxMER samples that have different levels of noise and different center of MER values. When there is not enough data or when the real samples don’t have much variability, we may use the synthetic data generator to provide a larger sample space for the classifier to get trained. With experiments, we have proven that the synthetic data acts very similarly to the real data during the training process and successfully prevented overfitting of the classifier, as the feature extraction can abstract the most important information from anomaly events, and get rid of the non-real parts from the synthetic data.

5.6. Classifer (Model)Training

The fundamental goal of machine learning is to generalize beyond the examples in the training set. This is because, no matter how much data we have, it is very unlikely that we will see those exact examples again at test time.

In Decision Tree Learning, a new example is classified by submitting it to a series of tests that determine the class label of the example. These tests are organized in a hierarchical structure called a *decision tree*. The training examples are used for choosing appropriate tests in the decision tree. Typically, a tree is built from top to bottom, where tests that maximize the information gain about the classification are selected first.

Support Vector Machine (SVM) is a supervised machine learning algorithm which is mostly used in classification problems. Each data item is a point in n-dimensional space (where n is number of features) with the value of each feature being the value of a particular coordinate. Classification is performed by finding the hyper-plane that differentiate the classes well.

In this paper, we chose Convolutional Neural Networks (CNN) as the basis for our machine learning model. (See reference [5] for details on CNN).

5.6.1. Neural networks & Deep learning

Neural Networks are models that are inspired by the structure and function of biological neural networks. They are a class of pattern matching algorithms that are commonly used for regression and classification problems. Deep Learning methods are a modern update to Neural Networks that exploit abundant cheap computation. They are concerned with building much larger and more complex neural networks. Deep Learning is a type of Neural Network Algorithm that takes metadata as an input and processes the data through a number of layers of the non-linear transformation of the input data to compute the output. Deep Learning is about learning multiple levels of representation and abstraction that help to make sense of data, the algorithm automatically grasps the relevant features required for the solution of the problem. In Deep Learning Neural Network, each hidden layer is responsible for training the unique set of features based on the output of the previous layer. As the number of hidden layers increases, the complexity and abstraction of data also increase. It forms a hierarchy from low-level features to high-level features. The most popular deep learning algorithms are: Deep Boltzmann Machine (DBM), Deep Belief Networks (DBN), Convolutional Neural Network (CNN), Stacked Auto-Encoders.

Convolutional Neural Networks are a category of Neural Networks that have proven very effective in areas such as image recognition and classification. ConvNets have been successful in identifying faces, objects and traffic signs apart from powering vision in robots and self-driving cars.

5.6.2. Prediction / Signature Recognition

The feature space is very much reduced and most of classification algorithms can handle it well. Although we can use SVM or KNN instead of using Neural Networks and they will work just as well as NN does for our current problem, we still use the deep architecture (Neural Networks) that offers statistical scalability, computational scalability and human-labor scalability over shallow architectures (KNN, SVMs..) to prepare for the increasing of the data amount and complexity in the future.[3]. The classifier we use now is a 6-layer convolutional Neural Network implemented in Keras with tensorflow as the backend.

We are using two separate classifiers, one to identify tilt issues, as they are across the whole CM RxMER sample, and another classifier to identify all other anomalies. The data is split up as 70 % for training and the remaining 30% as test. The neural network is using a batch size of 200 (number of samples). Within each epoch, the model is saved and retrained. The idea is to train the model, and then plug it back in and see if it really works. We are running 36 epochs until we reach the required training accuracy, i.e. the training loss per class. In each epoch we also validate against the Test data set, for accuracy (validation loss) against the test data. During each epoch, the accuracy increases. We stop training when the testing loss starts increasing.

5.6.3. Save Model (preservation)

Once the classifier gets trained and the model gets preserved, we add an additional stage to the anomaly detection process to use the classifier to identify what the anomalies are likely to be. Therefore, we can collect the information / ratings provided by the anomaly detector with details about the impact, location (on frequency), width, and group (signature recognition) of each anomaly happening on each CM, and this information can be used for higher level clustering and root cause analysis.

5.7. Event signature Identification

For all new test data incoming from the network we perform the following steps.

- Anomaly detection : As a preprocessing step, we use the anomaly detector to pick out outliers in the test data.
- Prediction / Identification / Labelling: The anomalies detected in the above step are sent to the Neural network model, to classify it as a particular type of issue. The anomalies are now labeled as a specific type, along with (subcarrier) location of the anomaly, the width etc.

The below sets of figures show the labels generated by the machine learning algorithm.

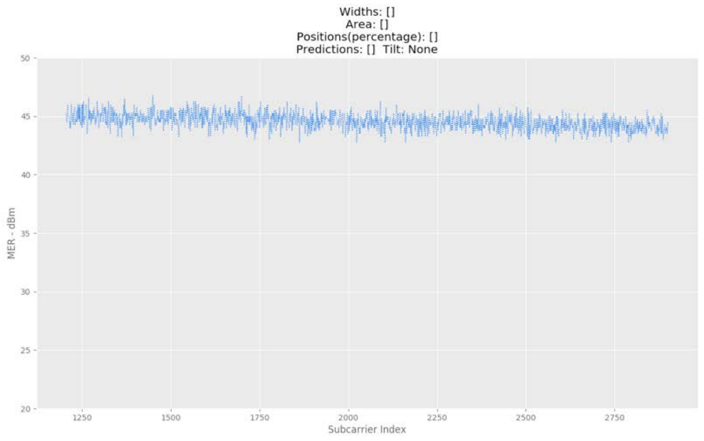

Figure 10 – Clean CM data, no issues identified.

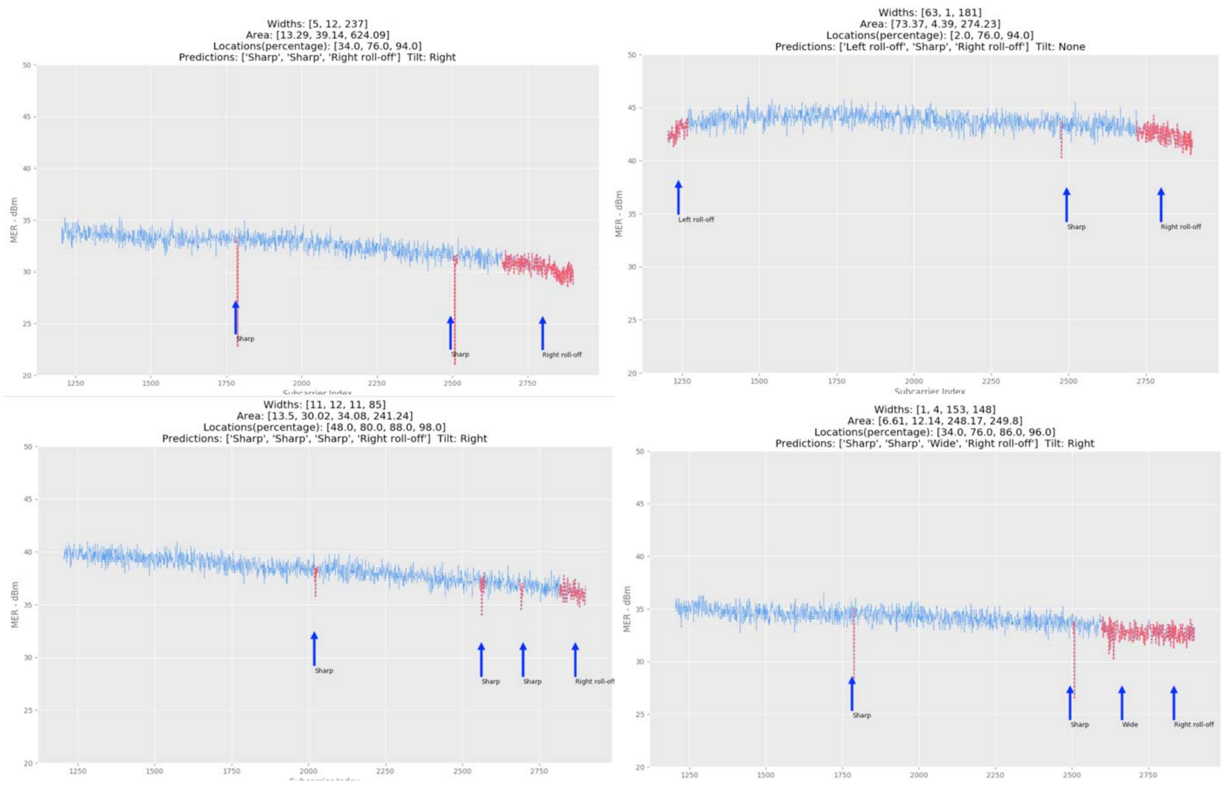

Figure 11 – Issues Identified: Tilt, Roll off

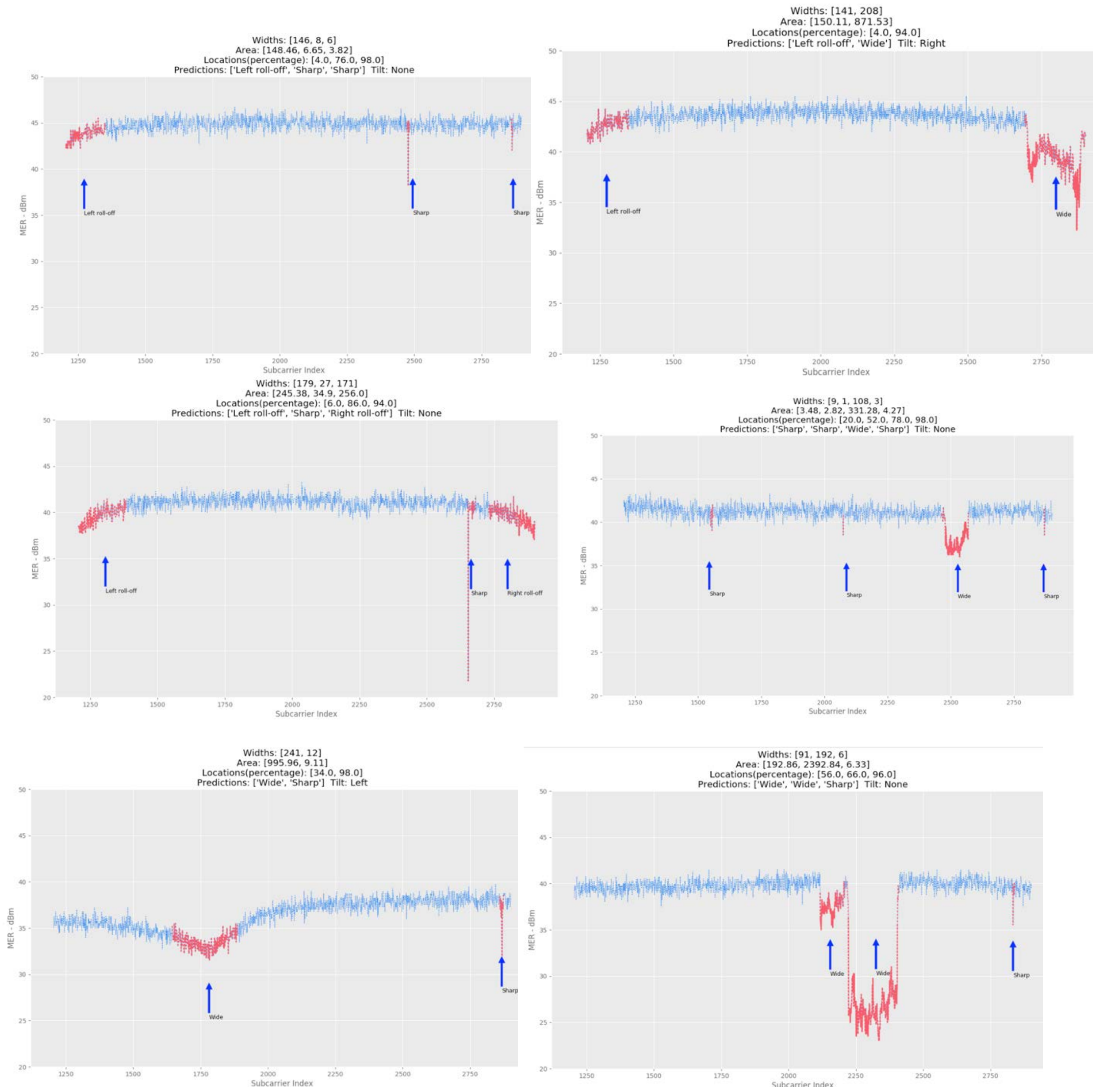

Figure 12 – Issues Identified: Wide dips, Sharp dips

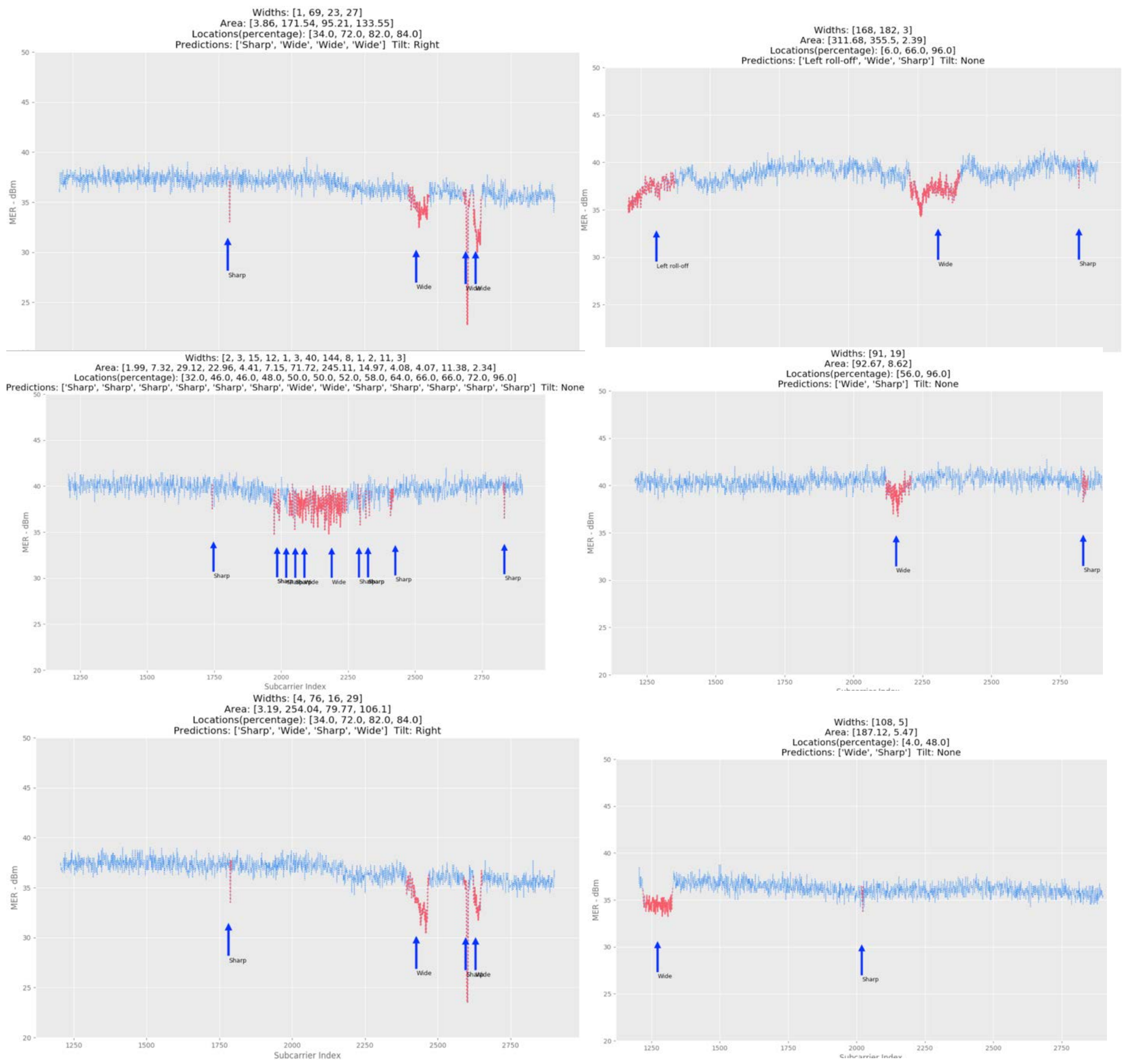

Figure 13 – Issues Identified: Various

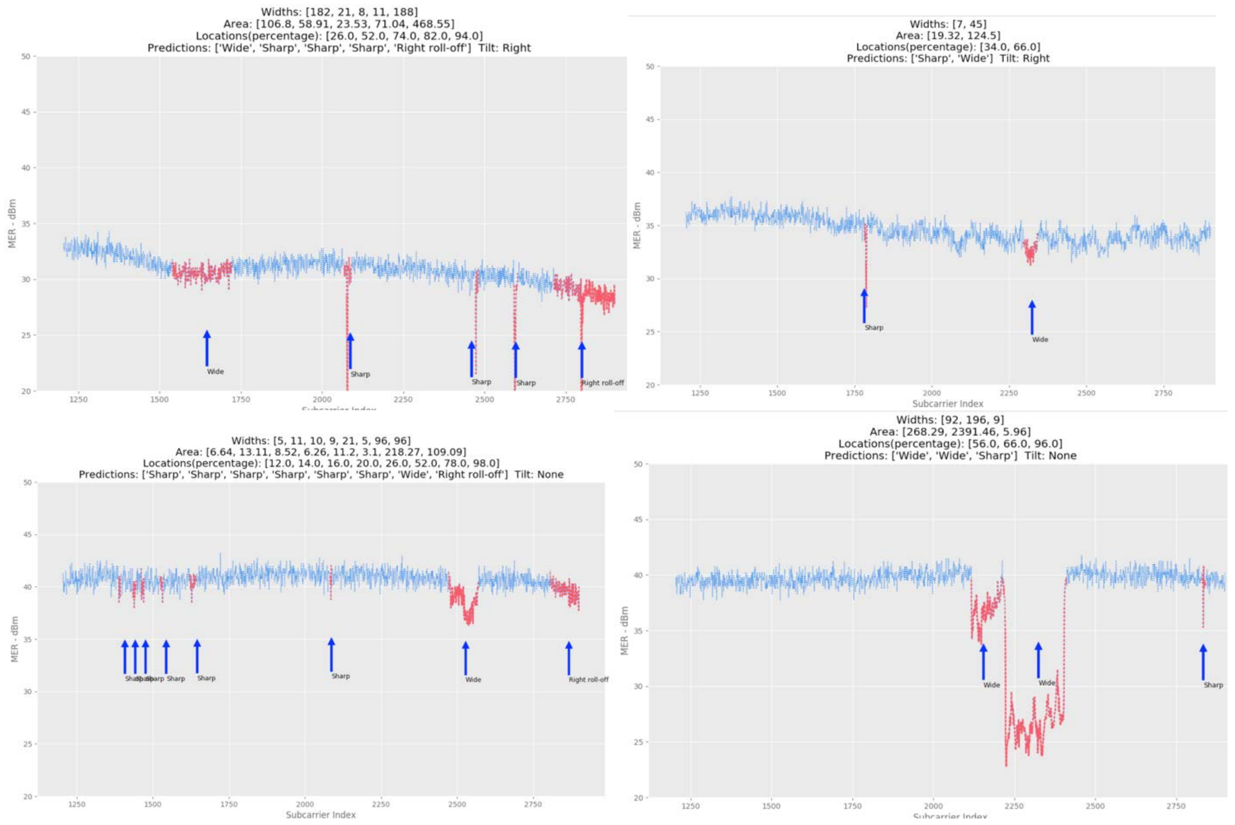

Figure 14 – Issues Identified: Various

5.8. Opensource tools used

There are many opensource tools can be used to test or create machine learning algorithms. Scikit-Learn is a convenient library that can be used to experiment with machine learning algorithms easily. We use Scikit-Learn to test the performance of different algorithms and the efficiency of our feature extraction algorithm. There are limitations of Scikit-Learn that it is not very scalable and not able to use the power from GPUs when training Neural Networks, but it's still a very good tool to do experiments with.

For loading and writing files, we use pandas. It's a very well developed opensource data analysis tool and it can be used to perform data preprocessing.

Instead of sticking with Python's matplotlib, we use Plotly to create plots and perform data visualization for the most part. Plotly is very powerful and easy to use across multiple languages such as Python and Javascript. We use Plotly to generate interactive 2D plots, 3D plots to help our data analysis process.

For constructing our neural network, we use Keras with Tensorflow as it's backend. Keras is a very useful library that abstracts the implementations of powerful machine learning libraries such as Tensorflow and Theano to help researchers and developers to easily and quickly create and test different structures of Neural Networks. The limitation of Keras is that it’s only able to use the power from a single GPU. This limits it’s usability when implementing large Neural Networks with large datasets. Some other libraries would support to utilize multiple GPUs to have shorter training time, e.g.: Theano. Those opensource libraries can be used in the future when the architecture of Neural Networks is mature and we need to deal with datasets that are very large.

6. Identifying global patterns

The Machine Learning classifier above gives us quick identification of issues across the DOCSIS 3.1 OFDM channel. The next step is to agglomerate these individual CM issues across all the CMs on the channel. Looking at the issues as seen by all the cable modems will give a view into the health of the network and the plant.

- Across topology
    - One view when looking at the data as a whole would be to identify issues across the plant topology. For example, the algorithms may be able to identify that many of the issues have a common root at some location in the cable plant.
- Across a specific geolocation
    - Another way to look at data as a whole would be to identify issues which correlate to specific geographic locations. For example, the algorithms may be able to correlate the issues seen in certain cable modems in certain geographic locations to say the LTE Channel map and frequencies for that location.
- Across CMTS
    - Another way of grouping issues in the data would be to group issues based on the fiber node are the CPS to which the modems are associated with.
- Across CM types
    - Another view of the issues could be to group issues seen by specific modem manufacturers together. There may be patterns which can emerge along this feature as well.

Root cause analysis: Analyzing the data across these global plant views, gives the operator a chance to understand the root causes of various issues. A simple histogram of the issues seen across a channel group by frequencies would be an indicator of interference issues, or other plant issues, on those sets of frequencies.

The figure below shows the simple proof of concept in which we are building a cable plant with modems associated with the right hierarchical elements in the network to topology. Once those issues in each of the modems are identified, we start looking at issues across groups of modems and present in various views the issues overlaid on the network map.

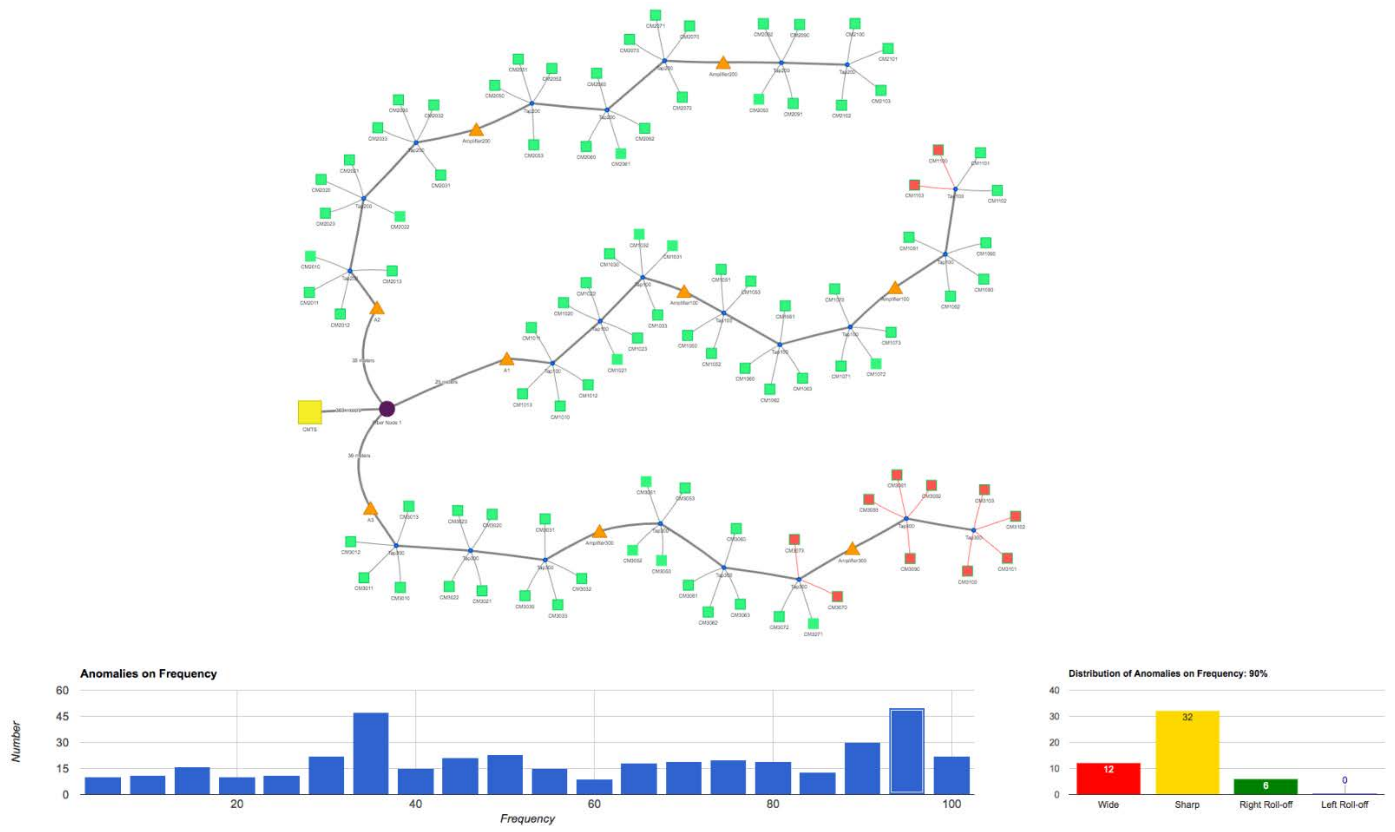

Figure 15 – Issue visualization, based on anomaly detection and pattern recognition

Conclusion

This paper presents the machine learning approach to automatically identify issues in the cable plant. Machine learning classifiers can quickly identify issues on new data samples based on previously seen anomalies. Starting from PNM data such as RxMER Data, we can identify anomalies which we use to train a Machine learning classifier. Once trained a neural network can reliably identify anomalies in new test data. These anomalies form a base layer of information about the plant. Consolidating data across multiple modems gives the operator a very good view into the various issues affecting the cable plant

Acknowledgements

Many thanks to Paul Schauer (Comcast) and Nader Foroughi (Shaw) for sharing data captures from their DOCSIS networks. Without the data they so graciously shared, this analysis would be based on synthetic data, which would be only be -3dB as much fun.

Abbreviations

bps	bits per second
CM	Cable Modem
CMTS	Cable Modem Termination System
FEC	forward error correction
HFC	hybrid fiber-coax
Hz	hertz
RxMER	Receive Modulation Error Ratio

Bibliography & References

Applications of Machine Learning in Cable Access Networks, INTX 2016, Karthik Sundaresan, Nicolas Metts, Greg White;(CableLabs;) Albert Cabellos-Aparicio (UPC BarcelonaTech).

A Comprehensive Case Study of Proactive Network Maintenance, SCTE 2016, Larry Wolcott, John Heslip, Bryan Thomas, Robert Gonsalves

Scaling Learning Algorithms towards AI, 2007 , Yoshua Bengio and Yann LeCun, http://yann.lecun.com/exdb/publis/pdf/bengio-lecun-07.pdf

How to Prepare Data For Machine Learning, by Jason Brownlee http://machinelearningmastery.com

An Intuitive Explanation of Convolutional Neural Networks, 2016 by Ujjwal Karn. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Filtering and Smoothing Data https://www.mathworks.com/help/curvefit/smoothing-data.html

[DOCSIS PHYv3.1] DOCSIS 3.1, Physical Layer Specification, CM-SP-PHYv3.1-I11-170510, May 10, 2017,Cable Television Laboratories, Inc.