SCTE•ISBE CABLE-TEC EXPO® VIRTUAL EXPERIENCE » OCTOBER 12-15 2020

2020 Fall Technical Forum SCTE·ISBE • NCTA • CABLELABS®

Using Machine Learning To Automate Node Split Designs And HFC Augmentation Options

A Technical Paper prepared for SCTE•ISBE by

Keith R. Hayes
Chief Executive Officer
IMMCO, Inc.
12395 Morris Rd, Alpharetta, GA 30005
770-378-3595
Keith.hayes@immcoinc.com

Table of Contents

List of Figures

1. Introduction

- **>300,000** Nodes
- **~700,000** Power Supplies
- **~2,000,000** HFC plant miles
- **>70,000,000** US MSO Internet Consumers
- **>40% CAGR** Broadband Consumption
- **Streaming video** here and growing, **online game-streaming** and **AR/VR** coming
- Oh, by the way, **COVID-19** induced **telework** and **virtual classroom** data activity

There are more than 300,000 HFC nodes in the US currently, and several million more worldwide. The additional network traffic triggered by Covid-19 in the spring of 2020 increased the level of HFC capacity augments by as much as 300% compared to 2019 volume. Network augmentation techniques such as node splits, adding HSI EIA's (6 MHz channels), service group de-combines, bandwidth expansion, Node+0/RPD's and mid-/high split reverse path expansion require engineering and operations resources in both ISP and OSP.

This paper will examine techniques in which much of this activity can be automated and iteratively optimized through Machine Learning (ML). With inputs provided from network mapping systems, capacity monitoring platforms, spectrum management applications, and business rules such as preferred augmentation hierarchy, expected duration before next augment, balancing of house-counts, municipal permitting difficulty, and cost efficiency, the ML environment would rapidly analyze entire geographic segments of the network, and provide augmentation planning data including network design changes and BOM's for areas requiring immediate physical layer upgrades.

As the ML platform iteratively processes network geographies, it will learn from how past predictions tracked to current status and continuously adjust to optimize capacity augmentation methods and designs. As the ML environment will be analyzing the entire network geography, data to drive Capex planning for future years will be derived enabling the operator to more efficiently allocate capital.

The ML environment would also support "what/if" network topology planning for approaches, such as bandwidth expansion to 1.8 GHz vs Node+0 at current bandwidth, and provide data on cost, duration before next augment needed, and percentage of network elements requiring replacement or repositioning.

2. Artificial Intelligence (AI) and Machine Learning (ML)

There is a lot of interest in using advanced computing technology to automate and accelerate processes that when done by humans are slow and tedious – facial recognition for example.

AI and ML are often used interchangeably but as depicted in the figure below ML is a subset of AI – so how are they related?

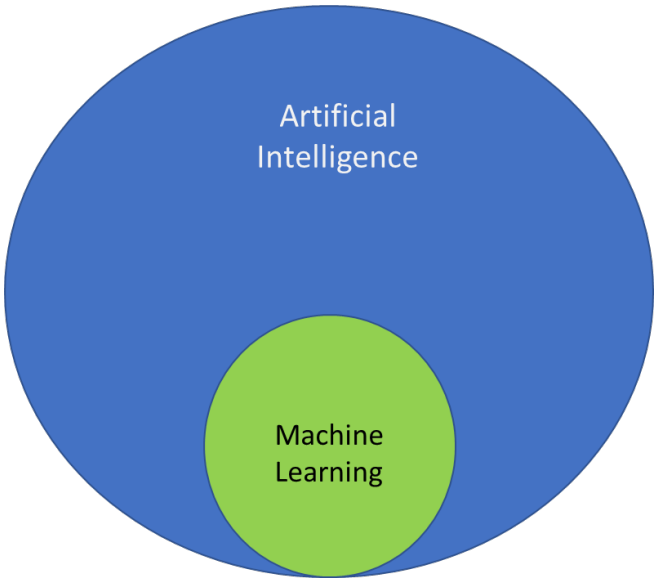


Figure 1 – Relationship of Artifical Intelligence and Machine Learning

Artificial Intelligence (AI) can best be understood as developing computer systems to perform tasks that humans can typically do better. A mechanical example would be a robotic welder in an auto manufacturing plant. A data analytics example would be Recommendations from your online retailer or streaming service that takes your past history and analyzes it against current inventory to "Recommend" items you may want to purchase or content you may want to watch.

Machine Learning (ML) is a subset of AI, and in addition to enabling computer systems to perform tasks with humans can typically do better, deploys "learning algorithms" that allows the system to measure performance and automatically improve outputs from those learnings. Back to one of the previous examples, if the robotic welder was coupled with an x-ray or gamma-ray measurement system that provided data on weld quality which then would change the parameters of the welding programming to improve performance without human intervention would be a good example of ML.

3. Network Capacity Augmentation Methods

One of the unique benefits of HFC architecture is the many options potentially available to add capacity, with more being developed as technology advances. The list below is not exhaustive but notes most of the typical options to add capacity without changing the bandwidth or US/DS ratio of the network, organized by rough order of difficulty to implement:

- Activate unused / repurpose EIA's (channels)
- Service Group de-combining/recombining
- Increase modulation density
- Extend fiber to heavy-use households
- Node Splits
    - Add transceivers to existing node (segmentation)
    - Add new node – minimum construction, no HHP balancing

  - Add new node – balance HHP
  - Add new node – balance for peak data utilization
- Add Remote Phy-Mac/Phy device

As evaluations are performed to determine the best approach on a node-by-node basis many other variables come into play:

- Municipal Permitting
- Maintenance Window availability
- Aerial/Underground

These many methods and variable are further complicated by the situation that most operators have separate departments managing ISP (Inside Plant) and OSP (Outside Plant) requiring careful coordination for those methods requiring both groups.

4. Implementing Machine Learning for Node Split Design

4.1. Programming Environment

When developing a Machine Learning application there are several fundamental environment decisions that must be made:

- Which programming language?
- Which IDE? (Integrated Development Environment)
- Which learning process – Supervised or Unsupervised?
- Which Decision Tree / Classification Engine?

4.1.1. Programming Language

Most common programming languages are capable of being easily used for ML environments, including:

- Python
- Java
- C++
- C#
- R
- JavaScript
- Scala

4.1.2. IDE – Integrated Development Engine

In order to provider an optimized development and debugging container, an IDE provides numerous tools and applications optimized for ML code development and testing. Many IDE's work with multiple programming languages but not with all, so analysis will need to be done to ensure the chosen programming language and IDE are compatible. Common IDE's include:

- PyCharm
- RStudio
- R-Brain
- Jupyter

- Spyder
- Geany

4.1.3. Learning Process

Machine Learning is either Supervised or Unsupervised. Supervised Learning is when the data being processed has been labeled and the ML system is taught by example, such as labeling pictures of items like a hammer and a banana so give the ML environment information to learn from. If the ML algorithm incorrectly identifies a shape, it would be "taught" by human input to improve its accuracy. Unsupervised learning is most useful in clustering and identifying similar objects and detecting significant anomalies – such as fraud detection identifying an abnormal financial transaction base on your spending history.

4.1.4 Decision Tree / Classification Engine

The last area that is needed to support the ML environment is a decision tree or classification engine. Common classifiers include:

- GBM Gradient Boosting Machine
- Random Forests
- Logistic Regression

The Python/Jupyter combination with Supervised Learning via random forest classifier was used for the development environment in which to build and test node split design via Machine Learning.

4.2. Network Data Extract

To provide data for the ML engine to examine it was necessary to extract every network element, both passives and actives, along with the HHP downstream/upstream, from the network map platform. As each element was geocoded distance calculations could be made without having to extract strand/cable/conduit data simplifying the extraction process. A table showing some of the parameters and results from a design iteration is shown below in Figure 2.

Equipment	Element ID	DS HHP	Leg Balancing DS	DS HHP Ratio %	Leg Balancing US	US HHP	US HHP Ratio %	Cascade Limit	Power Supply Proximit	Voltage	Current	Proper Signal Strengt	Minimal Construction	Room at Pole or Ped?
BTD-75SH AGC	5	523	NO	1	NO	0	0	YES	YES	50.56	1.5	YES	YES	YES
BLE-75SH	6	140	NO	26.8%	NO	383	73.2%	YES	NO	39.53	10.59	YES	YES	YES
9-TFC-4		133	NO	25.4%	NO	390	74.6%	YES	NO	35.58	9.5	YES	YES	NO
MB-75SH FD AGC	10	35	NO	6.7%	NO	488	93.3%	YES	NO	31.92	2.91	NO	YES	YES
MB-75SH FD	11	19	NO	3.6%	NO	504	96.4%	NO	NO	30.37	1.43	YES	YES	YES
BLE-75SH AGC	7	18	NO	3.4%	NO	505	96.6%	NO	NO	32.77	2.45	NO	YES	YES
MB-75SH FD	8	8	NO	1.5%	NO	515	98.5%	NO	NO	31.35	1.43	YES	YES	YES
MB-75SH FD	9	79	NO	15.1%	NO	444	84.9%	YES	NO	30.24	4.39	NO	YES	YES
BLE-75SH	68	47	NO	9.0%	NO	476	89.5%	NO	NO	28.89	0.99	NO	YES	YES
BLE-75SH	70	15	NO	2.9%	NO	508	95.5%	NO	NO	26.83	0.99	YES	YES	YES
BLE-75SH	69	16	NO	3.1%	NO	507	95.3%	NO	NO	27.32	0.99	YES	YES	YES
BLE-75SH	12	121	NO	23.1%	NO	401	75.4%	NO	YES	46.52	4.89	YES	YES	YES
BLE-75SH AGC	13	109	NO	20.8%	NO	414	79.2%	NO	NO	41.15	4.15	YES	YES	YES
MB-75SH FD	16	53	NO	10.1%	NO	470	89.9%	NO	NO	37.95	1.25	YES	YES	YES
BLE-75SH	15	10	NO	1.9%	NO	513	98.1%	NO	NO	38.37	0.86	YES	YES	YES
MB-75SH FD	14	34	NO	6.5%	NO	489	93.5%	NO	NO	39.72	1.2	YES	YES	YES
BTD-75SH AGC	21	203	YES	38.8%	YES	320	61.2%	YES	YES	51.75	4.59	YES	YES	YES

Figure 2 – Subset of Network Data Extract

Each element from the node to a line extender to a terminating tap was assigned a unique ID number and as the several performance parameters such as signal level, voltage, current and the ratio of HHP both

downstream and upstream from the network element were calculated. The node in the example above had 667 discrete network elements that were loaded into the ML environment.

4.3. Business Rules / Design Rules

After the network data was loaded Business or Design rules had to be created to guide the ML engine based on customer requirements. For example, an operator might want to design for the absolute least amount of activity needed to add capacity – adding another transceiver to segment a node without concern about balancing the homes-passed by the old node vs the new node. Another operator (or heck, maybe the same operator in another part of the network) might want to design for balancing the homes on the old and new node, and provisioning for future nodes. Some of the more than a dozen Rules included in this model were:

- Can the node be segmented?
- Is the proposed location geographically centered or not?
- Are homes passed balanced?
- Is cascade limit exceeded?
- Is there space on the pole or in the pedestal for the node?
- Does RF signal meet specifications?
- Does node meet minimal construction parameters?
- Is any coax reversing needed?
- Is Fiber Splice location reachable?

One of the benefits of creating an ML environment is it is relatively easy to add additional parameters and rules as technology changes, for example adding a rule to validate is power ampacity for small cell deployment was available, or tying the network data to the capacity monitoring system which could enable a split location to be balanced on peak data consumption versus geographic center or balanced Homes Passed.

5. Let's go split a node….with Machine Learning!

5.1. Machine Learning Node Split Design Process

In a standard Node Split design process, a designer looks at the network topology, and applying the Busines Rules using his or her experience picks a likely location for the new node. Afterwards, calculations have to be completed using a design tool such as Lode Data to verify the location will "work" – all signal levels are acceptable, there is appropriate voltage and no excessive current draw, and if those tests are passed the design needs to be completed and a BOM created. If the new node location does not "work", the process has to be repeated, each iteration taking 30 minutes or so for an experienced designer. The ML environment was developed to automate this process.

To test the new process several relatively simple node splits were loaded and processed by the ML tool, which generated network element relational schematics as shown in Figure 3 below:

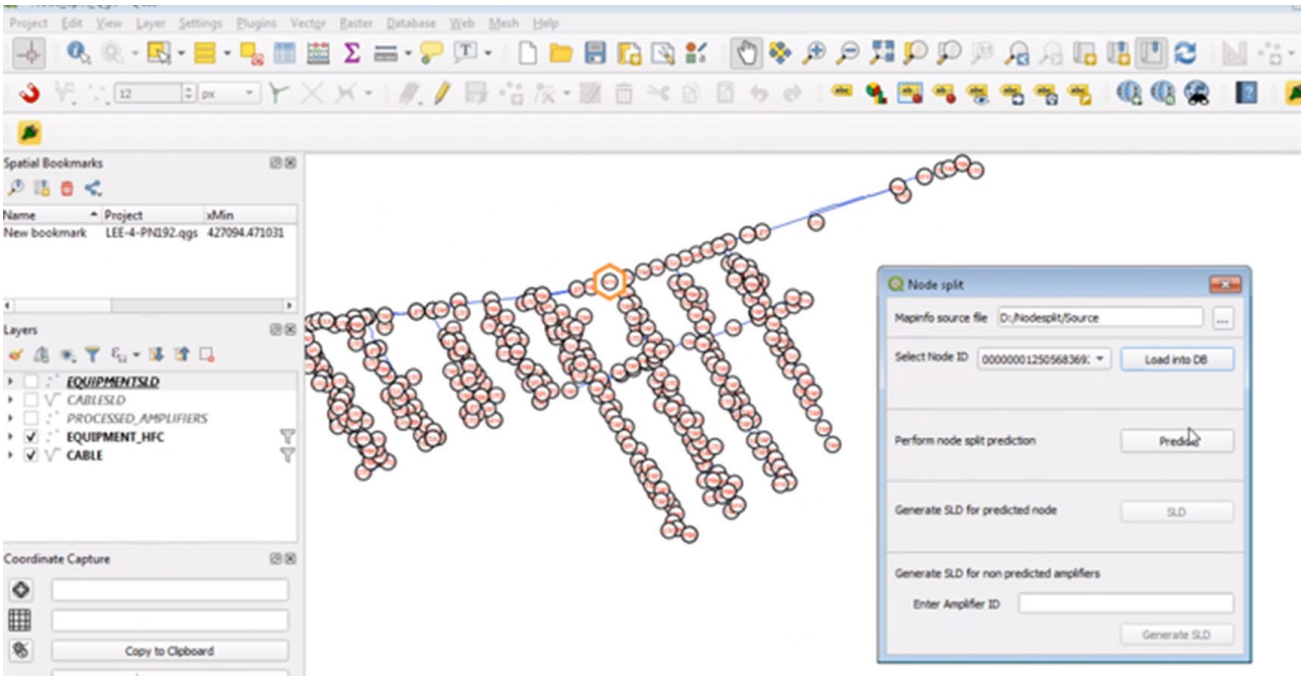


Figure 3 – ML Network Element Relational Schematic

The network element schematic is not needed directly by the ML tool but enables a designer to visually review the topology without using a network map system. Figure 4 is a close up showing the original node location and elements near it.

Figure 4 – Network Elements near the original Node

This particular Node was non-segmentable, and the Business Rules required minimal construction with no requirement to balance homes passed. The first iteration proposed the new node be placed on the fiber path to the existing node using spare fibers to connect it. As the new node location was an existing amplifer, there was no coax-resplicing required further complying with the minimal construction instruction – Figure 5 below:

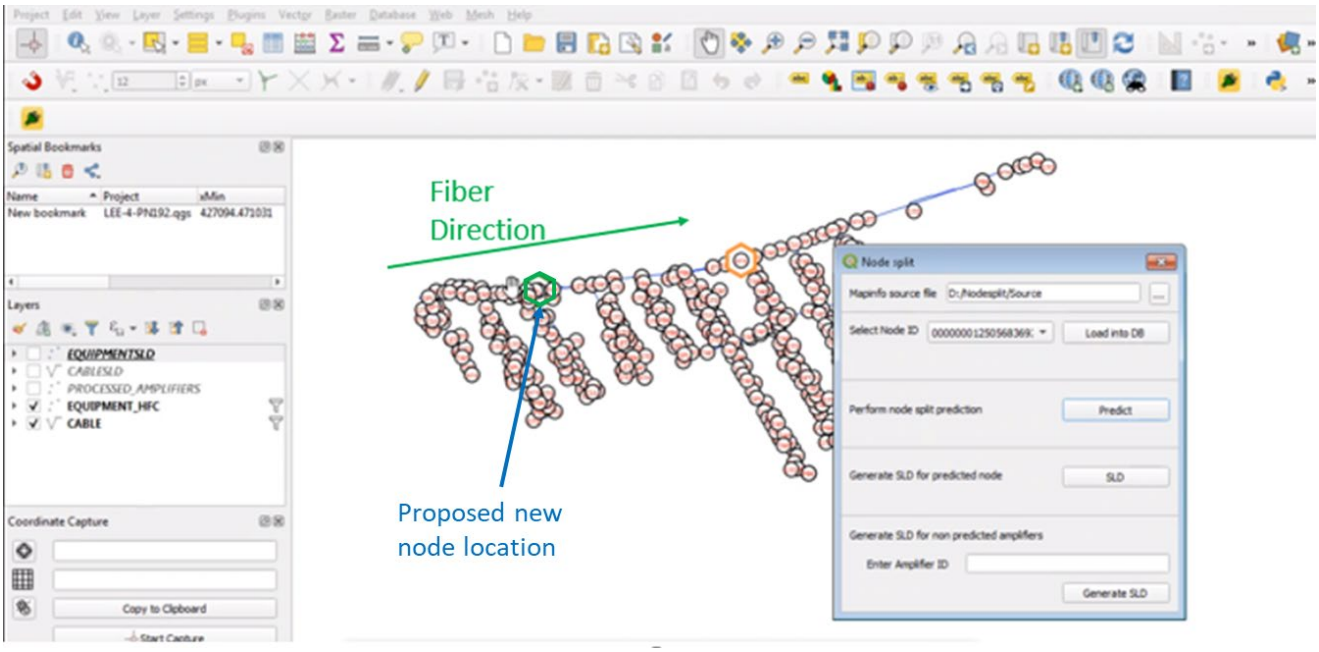


Figure 5 – New Node Location

When the households passed were calculated for the Original Node and New Node, there was definitely a mis-balance as shown in the table in Figure 6 below, but the operator would be able to provide additional capacity with little cost, time, or likely municipal permitting.

	HHP	Ratio
Original Node	457	100%

Original Node after split	329	72%
New Node after split	128	28%

Figure 6 – HHP after minimal construction Node Split

To "train" the ML environment, a dozen nodes of varying complexity, size and geographic orientation were used. Each node was "designed" manually with optimum location and business rule adherence, and several deliberately incorrect designs were created for each node configuration. The ML engine "learned" by comparing the correctly designed node split to ones that were incorrect.

After several iterations of supervised feedback, the ML node split algorithm was tested in 200 locations and determined the optimum location in 196 – an accuracy rate of 98%! Even more noteworthy was each location was processed in seconds versus around a half-hour for a human designer.

Below in Figure 7 is an example of a geographically complex node where the ML business rules required HHP balancing, future node segmentation capability, and construction costs and difficulty were not considered.

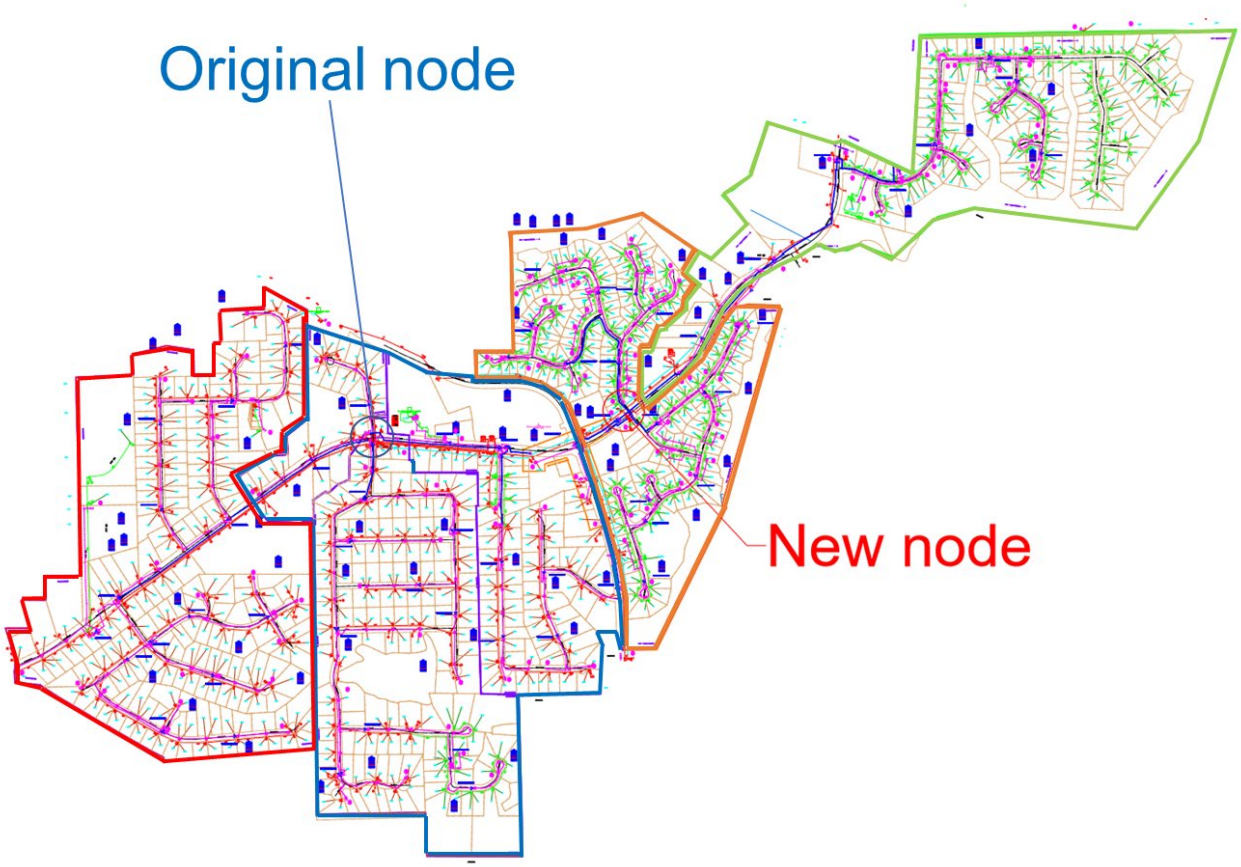
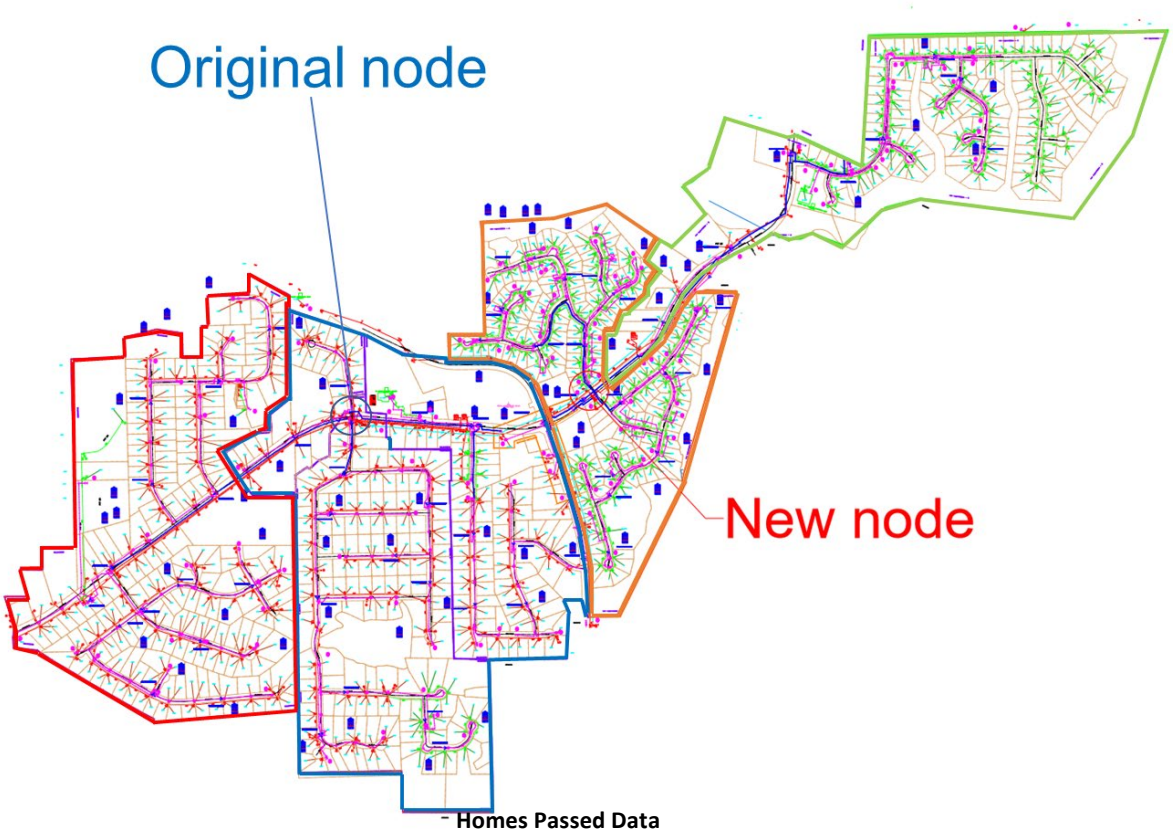


Figure 7 – ML – Designed Node Split with balanced HHP

This node served 523 HHP before the split, and after the split the ML design determined four legs each of which could be segmented into a node in the future with no construction needed, however more than 2,000 feet of fiber and extensive coax-reversing would be required. The HHP data is in the table in Figure 8 below:



Original Node	523

	Original Node Leg A	Original Node Leg B	New Node Leg A	New Node Leg B	Coax Reversing?	Fiber Extension Footage
Balanced HHP	111	136	178	98	Yes	2284
HHP % / leg	21%	26%	34%	19%		

Figure 8 – ML-designed Node Split HHP data

With good HHP balance and the ability to quickly double capacity via node segmentation this serving area will not require significant capacity upgrade activity for several years at nominal HSI bandwidth growth rates.

5.2. Implementing and Scaling Node Split Machine Learning Environment

The benefits of implementing a Machine Learning Node Split Design Environment are numerous and give the operators the analysis capability that would be cost-prohibitive in a standard design environment. For example, the nodes that are currently on the high-contention report could be quickly run through the ML design tool, under different business rules, and a Capital Expense projection could be created to guide

the budgeting process, and to inform the business about the tradeoffs involved where minimal construction augments result in more frequent network touches.

One challenge in implementing is the ML platform requires local data access – if your network is using a centralized network map platform such as GE SmallWorld or Synchronoss SpatialNet the maps would need to be "checked out" for data extract to the ML environment.

6. Creating a Holistic Network Capacity Augmentation Environment

As Machine Learning is integrated into the Network Planning and Operations process and platforms, as more systems and data are connected much more comprehensive design and analysis could be accomplished.

The chart in Figure 9 below shows some example inputs that if connected to a ML Prediction Engine could provide the network operator with planning information that could be calculated rapidly on a per-node basis for the entire network without significant human effort.

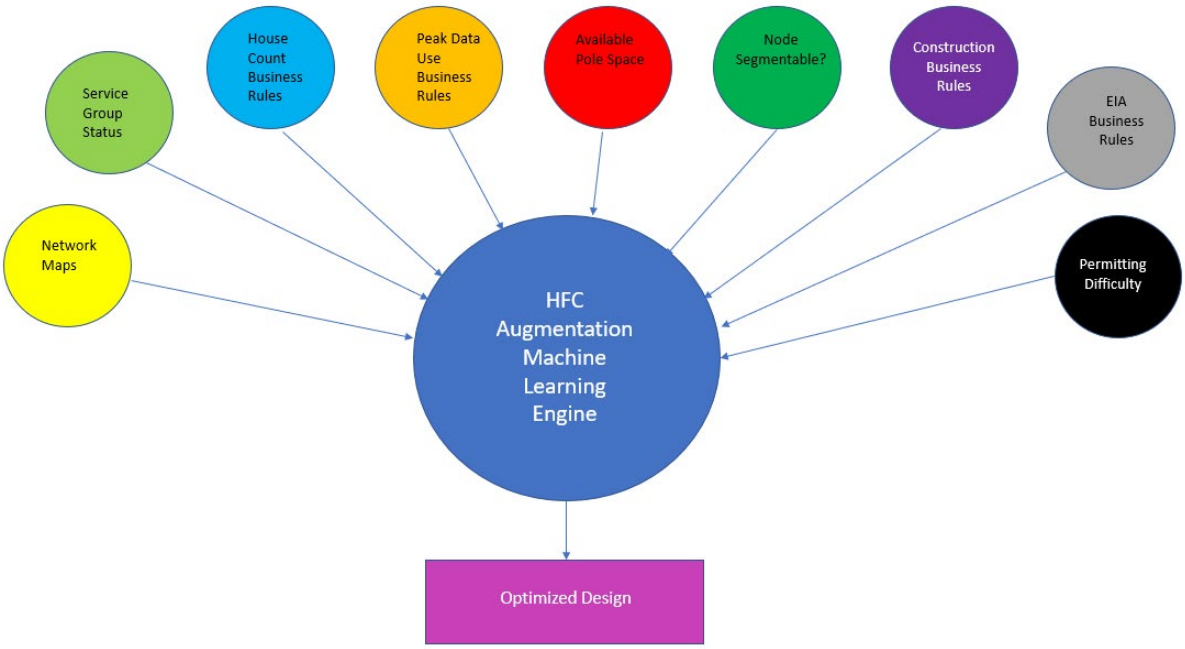


Figure 9 – Possible inputs to holistic HFC Analyis, Design, and Planning Engine

One possibility would be to connect the HSI network capacity platform and provide the ML environment data consumption information at a Peak-use-by-household basis. The ML business rules could be modified to optimize the node split based on real data usage versus just simply dividing the homes passed. A further business rule could identity the top 1% users and design and create BOMs for fiber connections to them to remove their data consumption from the HFC network – and cost scenarios could be quickly compared.

7. Conclusion

Network planning via Machine Learning is in its infancy but holds exciting promise to provide accurate and rapid automation of time-consuming network design activities. Network Operators should consider implementing network-based Machine Learning and over time connecting all network management platforms to enable comprehensive network design and capital expense planning.

Abbreviations

AI	Artificial Intelligence
AR	Augmented Reality
BOM	Bill of Materials
CAGR	Composite Annual Growth Rate
DS	Downstream
EIA	Electronic Industries Association
GBM	Gradient Boosting Machine
HFC	Hybrid Fiber-Coax
HHP	HouseHolds Passed
HSI	High Speed Internet
IDE	Integrated Development Environment
ML	Machine Learning
MSO	Multiple System Operator
US	Upstream
VR	Virtual Reality

Bibliography & References

Cloud Computing: Differences Between the AI and ML. https://www.comparethecloud.net/articles/cloud-computing-differences-between-the-ai-and-ml/

Machine Learning: An In-Depth Guide - Overview, Goals, Learning Types, and Algorithms https://www.innoarchitech.com/blog/machine-learning-an-in-depth-non-technical-guide

