CHALMERS
UNIVERSITY OF TECHNOLOGY

Integration of Machine Learning Algorithms into the Evolved Packet Core Network

Master’s Thesis in Complex Adaptive Systems (MPCAS)

Husam Starxin
Richard Lan

Department of Physics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017:06

Integration of Machine Learning Algorithms into the Evolved Packet Core Network

Husam Starxin
Richard Lan

Department of Physics
Energy and Environment
Chalmers University of Technology
Gothenburg, Sweden 2017

Integration of Machine Learning Algorithms into the Evolved Packet Core Network
Husam Starxin & Richard Lan

Supervisor:
Bengt Strömberg, BICP DUIC PC R&S MME System Test, Ericsson

Examiner:
Claes Andersson, Energy and Environment, Chalmers University of Technology

Master's Thesis 2017:06
Department of Physics
Energy and Environment
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: An antenna transmitting signals to various locations in the world, highlighting the networked society that could be enabled and augmented with the aid of machine learning techniques.

Typeset in LaTeX
Gothenburg, Sweden 2017

Integration of Machine Learning Algorithms into the Evolved Packet Core Network
Husam Starxin
Richard Lan
Department of Physics
Chalmers University of Technology

Abstract

In the world of telecommunications today, there is an increasing amount of data being transmitted to and from an increasing number of mobile phones. The systems and networks handling this data are therefore seeing the need of improvement in order to keep up with this evolution. Quickly becoming the norm for handling Big Data across several industries, machine learning is gaining influence and becoming integrated into the working practices of several world leading companies.

In this industrial thesis work we examine the Evolved Packet Core (EPC) network at Ericsson. This complex, distributed and real-time system handles the throughput of 4G as well as WiFi data to and from mobile phones. The purpose is to study the network with the idea of improving it using various machine learning techniques.

The thesis is divided into two parts, literature review and ideation. The literature review aims to reveal what has already been done in the telecommunications industry, with respect to machine learning. The ideation phase incorporates a full study of the EPC network, which includes reading technical documentation, interviewing experts on various subsystems as well as hosting workshops at the different departments of the company. Additionally, ideas are innovated with the newfound knowledge of the EPC network. The goal of each idea is to improve the EPC network by either adding new features or improving existing workflows.

The innovated ideas are categorized into areas such as subscriber profiling, performance monitoring and test analysis amongst others. The EPC network shows promising and untapped potential in terms of evolution incorporating machine learning. Finally, we also provide an outlook on the future of the telecommunications industry. The two main topics discussed in recent years in the industry are 5G and Internet of Things (IoT), to which there is potential for integrating machine learning as well.

Keywords: 3GPP, Ericsson, EPC, EPG, Machine Learning, MME, Networks, Telecommunications.

Acknowledgements

We would like to extend our gratitude to Ericsson and its Packet Core department for hosting our thesis. A special thanks to our line manager Bengt Strömberg for all the administrative help we received and the sheer amount of people he put us in contact with. His consistently, supportive approach to us and his surroundings really makes life easy to live around him.

Another thanks goes out to the other folks at Packet Core. This includes Anders Larsson P for playing a crucial role early on in helping us to build an architectural understanding of the EPC network, Patrik Willard for his accommodative and co-operative attitude and Tobias Englund in showcasing some exciting 5G and EPG things. We are also indebted to other sections for hosting us in various presentations, the managers of which encompass Karin Nordling, Niklas Zetréus, Fredrik Ivarsson and Jan Lundkvist. The employees of these sections ended up playing key roles in our interviews, when we were learning about the various nodes of the system.

The monthly machine learning meet-ups in the Garage arranged by Gordon Sun were rewarding as well, since we got to interact with other thesis workers performing related tasks. Lastly, a big thanks to friends and family for providing the emotional support we needed to carry this thesis work out to the end.

Husam Starxin & Richard Lan, Gothenburg, June 2017

Contents

1 Introduction

The telecommunication industry holds some of the largest and most complex systems in the world. These systems combine characteristics such as, real-time processing and communication, distributed computing as well as fault tolerance. They work with each other over large synchronized networks, where each system becomes an independent node or component in the network. Each node in the network generates large amounts of data that need to be processed, analyzed and reacted to immediately. Dealing with this much data is a costly process that takes a long time and it would therefore be ideal for companies working in this industry to be able to handle such large data through optimized and automated means.

This is where the concept of machine learning comes into play, as it is a powerful tool that can be used to automate some of the tedious tasks that involve dealing with large data. However the integration of machine learning algorithms and techniques into large and complex systems such as the ones used within the industry is not an easy task. It requires extensive research in order to establish gaps within the network where the algorithms can be utilized appropriately and effectively. There is also a need to understand the type of data that the various nodes of the network deal with in order to be able to use that data as parameters for the machine learning algorithms.

The purpose of this thesis is the study of such large scale networks, specifically the Evolved Packet Core network (EPC) and its various systems and sub-components in order to identify possible areas within the network where machine learning algorithms can be integrated in order to automate some of its tasks. The thesis involves the study of existing literature in order to identify viable machine learning methods that have been successfully integrated into telecommunication networks, and then evaluate whether they are applicable to be utilized within the EPC network. The thesis also involves the introduction of new features and improvements to the network which are also achieved through machine learning.

The intention is to explore both supervised and unsupervised learning, depending on the type of data used and tasks that are performed by the network's various components. This is done through extensive research of the network's main and support nodes, followed by detailed proposals of implementation of new and existing tasks within the node using machine learning.

Collaboration with Ericsson

The reason behind the focus of the thesis on the telecommunication industry is because it is a field where application of machine learning is a fairly novel concept. This decision was also motivated by the fact that we were able to find an ideal environment where we can do our research. The investigation, implementation and testing parts of the thesis will be done at one of the world's largest telecommunications service provider (TSP), namely Ericsson. Ericsson is not only one of the leading enterprises within the field of telecommunication, but also in networking, with a multitude of products being used for communication on a global scale. Most of Ericsson's products have full scale implementations of parallel distributed fault-tolerant and real-time systems. Their products are thus particularly useful for the scope of our thesis.

The particular product we will be basing our research on is the Evolved Packet Core (EPC) network [Eri]. It is a packet-oriented mobile data service for 4G and Wi-Fi cellular communication systems. It is made up of a number of components, which handle specific tasks. An example of a component is the "MME" node, which handles the registration and connection of subscribers or users in the 4G communication realm. Another example is the "EPG" node, which handles the transfer of user data over the Internet.

The EPC network is vast and contains vast amounts of subsystems that handle large quantities of user data. It is therefore an ideal environment for finding possible implementations of machine learning algorithms.

Significance of the Thesis

The results of this thesis are intended to be of value for both the industry and academia. This is because the new knowledge gained from this thesis could be applied in various types of software applications within the industry as well as solving problems within the fields in computer science and engineering. While machine learning is a relatively young and untested scientific field, it has shown great promise in terms of applications in other fields than telecommunications. Examples include image recognition, deciphering musical tastes and playing complex board games. Considering this wide range of applications, this research could have a great impact on both industry and academia.

Systems that utilize manual labor or manually programmed tasks can be costly or even unfeasible at a large scale. This is why machine learning might be an important factor of optimization for industries that have limited exposure to the field, whether it is a question of automating tedious tasks or handling assignments that would otherwise be computationally impossible for a human mind. From an industry point of view, this research can supply proof of the usefulness of machine learning and how it could be implemented.

From an academic point of view the potential benefits of the thesis would be to show another application area for machine learning, namely telecommunications. This would in turn shed a further spotlight on how useful machine learning as a scientific field could be for future academic research.

2

Theory

In the following sections, the necessary foundation needed for the rest of this paper is laid. Section 2.1 gives a comprehensive overview of the field of machine learning, with general descriptions of several existing categories within the field given in each subsection. Section 2.2 gives the reader an overview of the Evolved Packet Core network, as established according to the so called 3GPP-standard.

2.1 Machine Learning

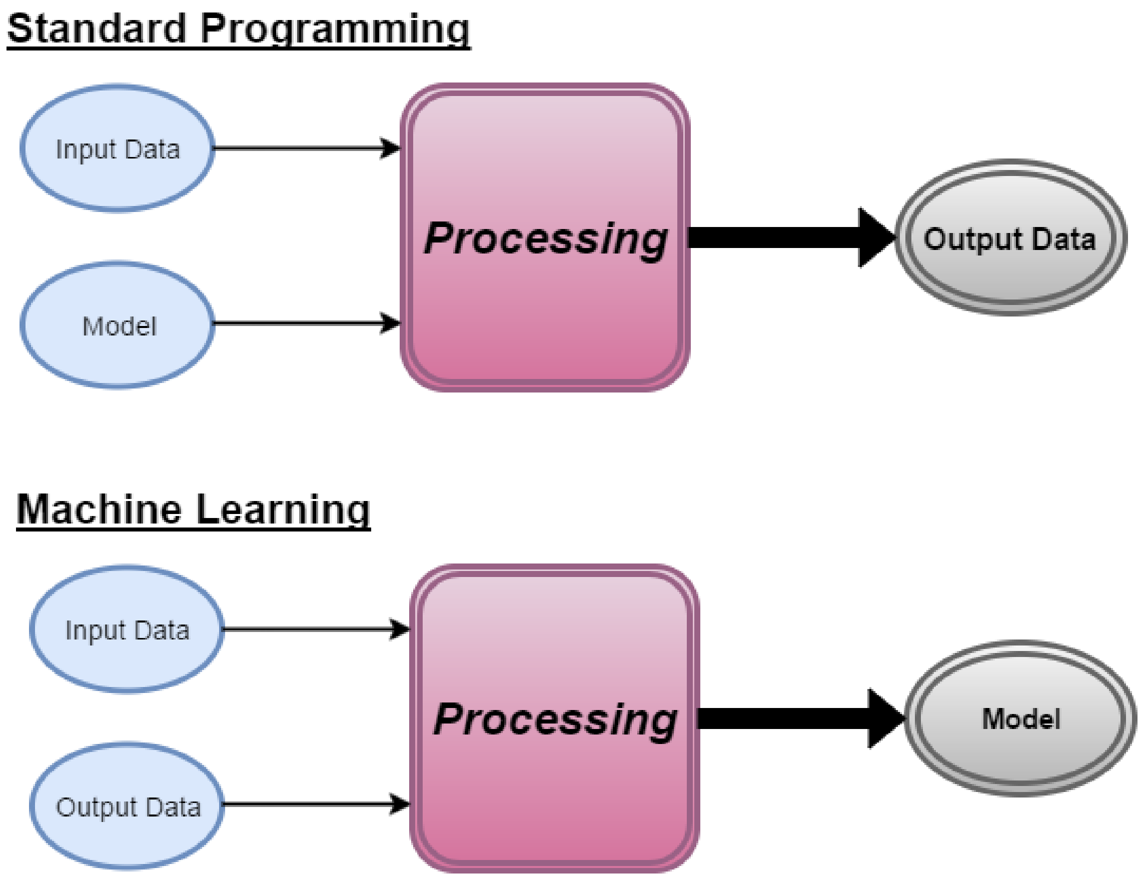


Figure 2.1: Highlighting the differences between standard programming practices versus machine learning.

In the book Machine Learning, written by Tom M. Mitchell, a definition for the scientific field of machine learning is given as "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.". The idea of machine learning is to automatically extract a program that can perform a certain task, based only on the data produced by a system. As can be seen in Figure 2.1, machine learning is comparable to traditional programming. However, there is a fundamental difference in the objectives. Whereas the goal in standard programming is to extract output from a machine that processes input and model, the goal with machine learning is to extract a model from a machine that processes inputs and outputs. The models can be of both black and white box nature. There are three classical categories of machine learning, namely supervised, unsupervised and reinforcement learning, each with specific tasks one seeks to have a computer perform. Primary examples include classification, regression and clustering tasks. These will be detailed further in the coming sections. [Mit97]

2.1.1 Workflow

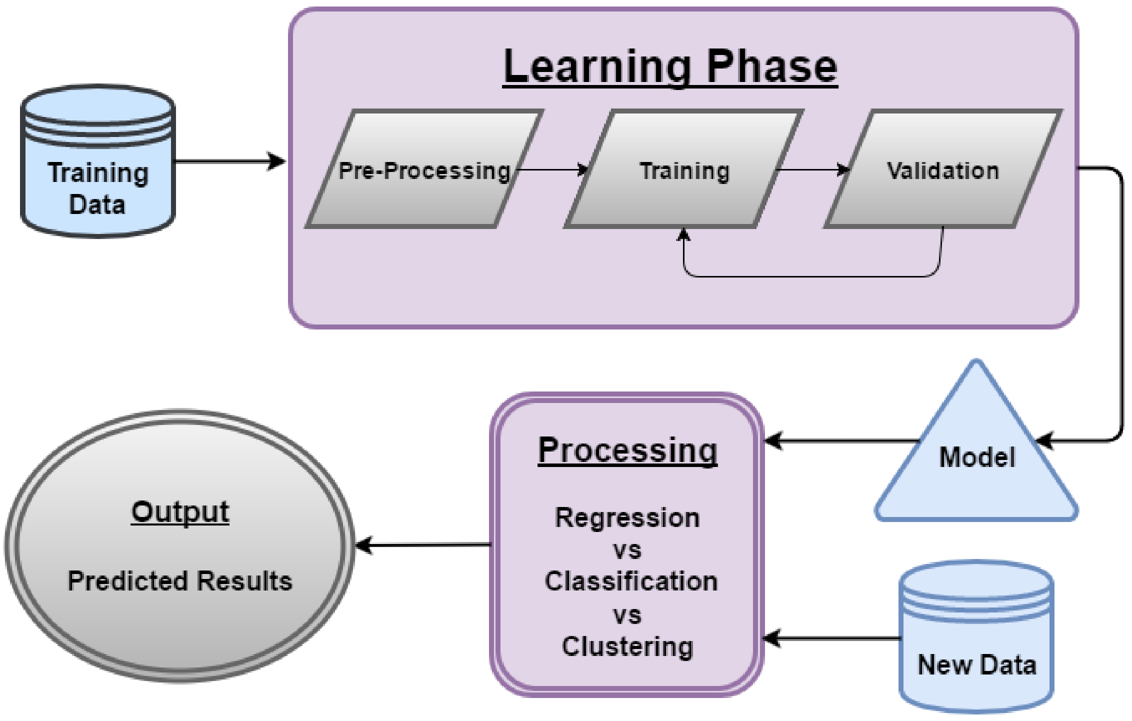


Figure 2.2: The standard workflow of a machine learning implementation.

In general, there are three main stages in creating a machine learning model, see Figure 2.2. Essentially, one needs to identify a problem one would like to solve and prepare the data one intends to use in pre-processing stage. After pre-processing comes the training phase. At this stage an appropriate algorithm needs to be chosen and then trained, so that it predicts the data it is being fed correctly. The extrapolated model then needs to be validated in order to obtain prediction accuracy and its ability to generalize to new data produced by the system. If the model achieves a satisfactory prediction accuracy in the validation phase, it can be considered ready for deployment. [Mit97]

Pre-Processing

After identifying a problem to be solved, or task to be performed, data needs to be gathered. From the data one must choose which variables to use as inputs for the machine learning algorithm, this is also known as feature selection. Feature selection primarily relies on the domain knowledge of a human expert and can heavily influence the performance of the model. If some features are superfluous or redundant, they can contribute with unbalanced weighting to the data, making certain patterns more important than others, or they can contribute with noise, making it more difficult for the algorithm to learn the proper patterns in the data. Similarly, if not enough data points for training exists, the model will find it more difficult to generalize to data outside of the training set. These phenomena would exhibit itself with a sub-optimal prediction accuracy in the validation phase.

The machine learning pre-processing does not fully need to rely on human intervention however, there are several tools that can augment the feature selection. One example is using a kernel trick in order to enable basic linear classification algorithms to perform more advanced non-linear classification tasks. In essence, the kernel trick implicitly maps the input vectors to vectors of higher dimensionality, thus simplifying classification.

Another tool is the Principal Components Analysis (PCA), useful when one wants to reduce the number of features. PCA yields a similarity measure on the features and can in this way inform whether several features in the data are redundant or not. The superfluous features can then be removed from the data altogether, thus reducing training time for the machine learning algorithm.

Training

When the pre-processing phase has been completed, it is time to select an appropriate algorithm to deploy in the model for solving the task at hand. Each type of task usually comes with an associated class of algorithms. If the task is classification for example, then several algorithms exist for solving this task in particular. Examples for this type of task include decision trees and perceptrons, some of which will be detailed further in later sections.

An important aspect of the training phase is parameter tweaking. As an algorithm has been chosen, it comes with a number of parameters that have various effects on prediction performance. General for most algorithms within the classification task are learning rate and training time, usually defined by number of time steps or training epochs. If one is using a multi-layered perceptron for example, which is a type of a neural network, it comes with parameters such as number of hidden layers and number of nodes within each layer. It is therefore recommended that one commits to training several versions of the same algorithm, each with slightly tweaked parameters, in order to obtain optimal performance for the model one is attempting to create.

Training refers to how the algorithm updates itself in order to improve performance in prediction. Normally, the initial prediction model is completely randomized. With each passing time step, the intention is for the model to improve slightly so that it predicts more accurately. Hence, we have a computer program whose "performance at tasks in T, as measured by P, improves with experience E".

Termination of training can be set in several ways. One way is to set the number of epochs, meaning how many times a data point in the training set would on average be used for training, as a termination criteria. Another would be to use how much the prediction accuracy has improved over a finite number of the most recent time steps or some other form of validation against the training set.

In the earlier mentioned multi-layered perceptron example, input data is fed into input nodes at each time step and yield values in output nodes. Training then changes the values of interconnected weights, between input and output nodes, iteratively. These weights store the pattern information and essentially instructs the model what output to yield for inputted data.

One would typically use the error measure as the reference point for updating in supervised learning. If the task is a regression task, then a correctly predicted data point in the training set would not update the model, since correct predictions are desirable. Meanwhile, an incorrectly predicted data point yields some modification, so that the model predicts the data point more correctly next time. The extent of the modification is usually determined by the magnitude of the aforementioned error measure.

Validation

Finally, after the training phase, the model is validated against data outside of the training set. This data is usually referred to as the validation set. The idea is to control whether the final model obtained also manages to perform well on new data that is presented to it by the same system that produced the original training set. This is known as the ability of the model to generalize. If the model generalizes well enough, as determined by a human expert, it can be deployed into the system. Otherwise, the model returns to the training phase and re-trains until acceptable prediction accuracy has been achieved or the model is abandoned in favor for a different one. If ready for deployment however, the model is paired with new data produced by the system and processed for new results. These new results are essentially predictions of some kind, depending on the task it is intended to perform.

2.1.2 Categorical Division

As mentioned before, there are three classical categories of machine learning. The differences in workflow between each category is visualized in Figure 2.3. Supervised learning mainly follows the workflow as described in Section 2.1.1, with two examples of algorithms given. Unsupervised learning deviates slightly, since no output data is available the training and validation is performed in different ways, as described in Section 2.1.4. Finally, reinforcement learning has a completely different blueprint to the supervised and unsupervised learning approaches, which is described in Section 2.1.5. Both unsupervised and reinforced learning come with one algorithmic example each.

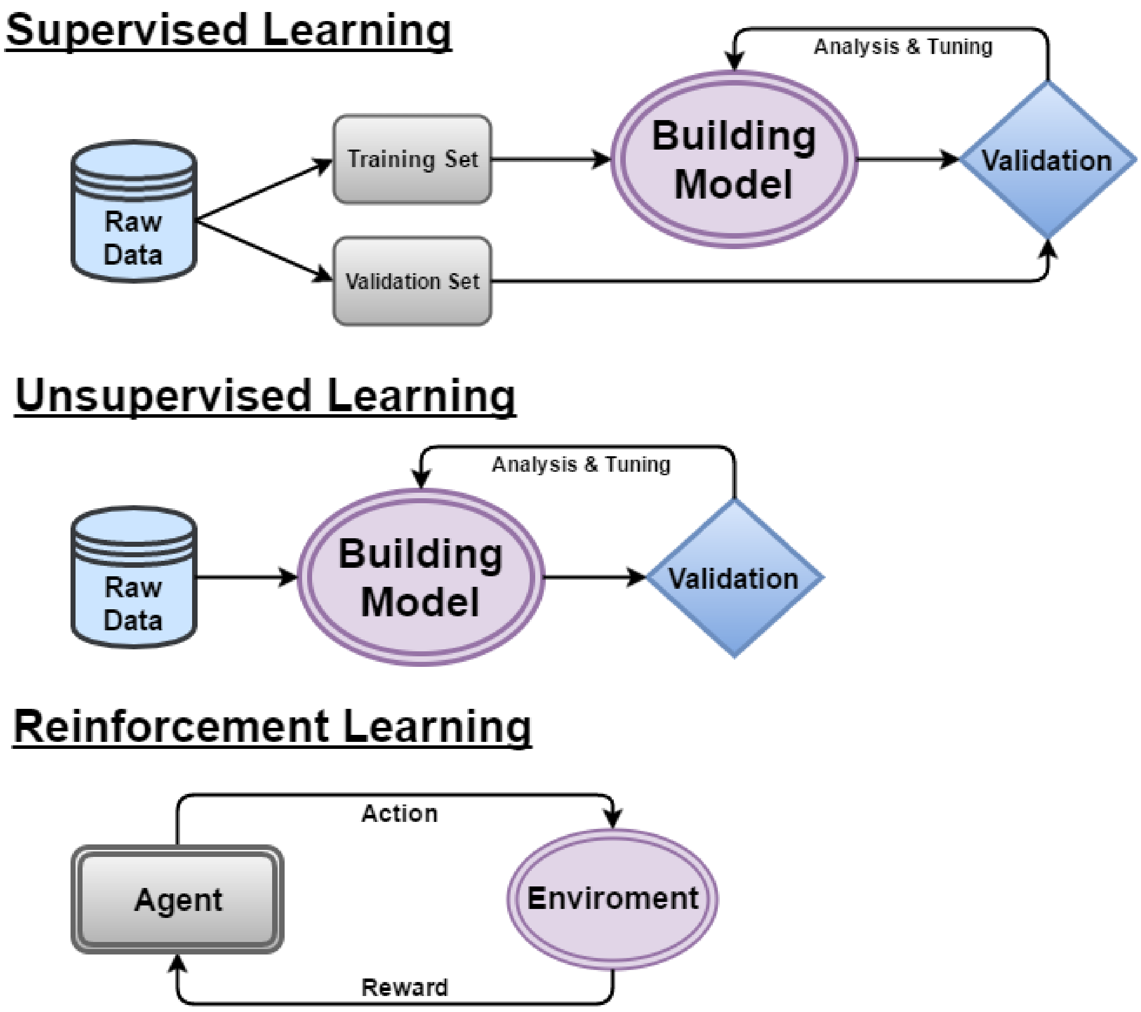


Figure 2.3: The three classical categories of machine learning. Here, the differences between the categories are highlighted by their workflow diagrams.

2.1.3 Supervised Learning

Supervised learning is one of the classical categories of machine learning and requires both input data and the correlated output data for said inputs. The reason for it being called supervised is because the outputs are used in the training phase to guide the training of the model, as described earlier. There are two types of tasks within supervised learning, classification and regression. [Mit97]

Classification is considered qualitative and means categorizing or labelling data points [Mit97]. A classical introductory example is the Modified National Institute of Standards and Technology (MNIST) database of handwritten digits. These are represented on images of 20 × 20 pixels, with the greyscaled color of the pixels indicated by real values. The pixels of the images are used as inputs while the output indicates which digit the inputted image represents. The goal is to correctly recognize which digit a handwritten symbol closest resembles. [LCB]

More generally, the objective of classification is to correctly label data points given to it. See the left panel of Figure 2.4 for a visual representation of an example that has two input features and one output variable for binary classification. The data points are classified according to color, with the dotted line representing the classifier itself. Ideally, the classifier will split the colored points into two regions, according to their respective classifications. [JWHT13]

Whereas classification has discrete outputs, regression has continuous outputs and is considered a quantitative approach. The objective here can be likened to mathematical function fitting, which can be seen in the right panel of Figure 2.4. [JWHT13]

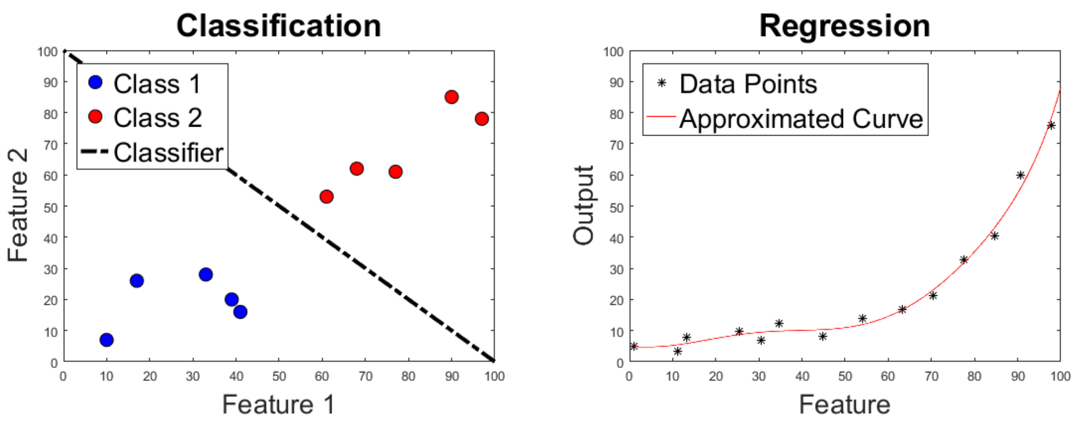


Figure 2.4: Exemplification of how the objective with the two supervised tasks, classification (left panel) and regression (right panel), can be visually represented.

Decision Tree

The Decision Tree (DT) is one of the most common and basic supervised learning techniques. It is used for classification, where the tree recursively splits into binary branches that classifies an inserted example. At each splitting point, or node, there is a logical question that tests an attribute or feature, and then sends the inputted example down further in the tree until it reaches the endpoint of the tree. [JWHT13]

There are several algorithms for selecting the appropriate question for each node, which is the learning or training part of the DT method. Most algorithms utilize a greedy search of all possible trees, starting at the top, and selects the optimal one. Iterative Dichotomiser 3 (ID3) and C4.5 are examples of these, developed by Ross Quinlan in 1986 [Qui86] and 1993 [Qui93]. Essentially, at each node selection, both ID3 and C4.5 will try to split the tree into two branches using a question from which most information can be gained. This is achieved by testing how well a feature separates the output classifications. Another question to consider when constructing DTs is the size of the tree. The inductive bias of ID3 favors shorter tree branches and is justified by Occam's Razor, cutting out superfluous parts of the tree. A smaller tree can typically also avoid overfitting the training data and generalize more accurately. To achieve this smaller type of trees, techniques such as pruning and growth halting exists. Quinlan himself describes a heuristic called the minimum description length principle [QR89] for stopping the growth of a tree. [Mit97]

The strength of the DT lies in the interpretability of its visual representation; it is easy to explain why the tree makes the infers it does. While the DT is fairly intuitive however, it also lacks predictive power in comparison to several other statistical methods, especially when there is noise in the data. To overcome this, aggregating results from multiple DTs can increase its performance. These methods include bagging, boosting and random forests. Bagging, for example, attempts to reduce variance in the data by averaging the outputs from a set of observations. In this case the observations would equate to a plurality of non-pruned DTs, each trained on a separate training sets. [JWHT13]

As previously mentioned, the DT is used for classification. There exists however a transformation of it that handles the regression task instead. This transformed tree is known as a Regression Tree (RT) and functions similarly to the DT, but has other criteria for termination. [Mit97]

Support Vector Machine

The Support Vector Machine (SVM) is another popular method of classification. The basic concept is to binary classify by splitting the data points with a hyperplane. A hyperplane is always of one dimension less than the space it operates in, such as a flat surface inside a three dimensional room. An example of a one dimensional hyperplane (or straight line) doing the separation of data points inside a two dimensional input space can be seen in the left panel of Figure 2.4. [JWHT13]

Since the hyperplane is always straight or flat, the classifier works optimally if the data is linearly separable. This means that every data point can be classified perfectly, as in the left panel of Figure 2.4. In this case, the maximal margin classifier is constructed. The goal here is to implement a hyperplane that separates all the data points into their respective classes, while maximizing the margin (or distances of the data points) from the hyperplane itself. This margin equates to the level of confidence of the classifier, a greater margin means a more confident classifier. However, the classifier is clearly quite sensitive to data inputted. [JWHT13]

However, the idea of the separating hyperplane can be extended by using a soft margin, meaning that the classifier allows for data points to enter the space between the margin and the hyperplane on both sides of the plane. This soft margin classifier is a generalization of the maximal margin classifier, known as a support vector classifier, and can also be applied to the case when data is not linearly separable. Here, the hyperplane almost separates the data points perfectly by allowing some data points to be on the wrong side. Using so called slack variables to account for the violating data points, the goal is still to maximize the aforementioned margin.

The process of maximizing the margin in the SVM algorithm can very much be seen as a linear optimization problem. However, the properties of the optimization problem implies that data points correctly labelled and outside of the previously mentioned margin space do not affect the classifier itself. Only data points inside margin spaces and incorrectly labelled data points affect the classifier and form what is known as support vectors. [JWHT13]

Another solution to a data set that is non-linear in its separability is to manipulate the input space. The inputs fed to the classifier could for example be raised by a power, meaning it is put into polynomial form, rather than fed linearly as one typically does. This non-linear type of transformations is the final step of generalization that results in the SVM and it comes from a specific method of manipulating the input space, called a kernel. A kernel consists of a function that measures the similarity between two data points, typically via the inner product of said data points. Popular kernels often used are polynomial or radial ones. The strength of the kernel method is that it does not need to work in an enlarged input space explicitly. The input space for radial kernels is in fact infinite-dimensional and only implicit, thus making the number of computations unfeasible. [JWHT13]

The concept of that SVMs are built on, the idea of the hyperplane, does not apply well to cases with more than two classes. There are however two attempts to tackle this issue. The first way of addressing this problem is called one-versus-one classification. The method incorporates constructing one classifier for each pair of classes in the data set, then assigning each data point in the test set to the class that it most frequently lands in. The second approach is called one-versus-all classification and, as the name suggests, creates one classifier per class that distinguishes data points that do belong to the class and those that do not. The final step involves assigning data points in the test set to the class for which the margin is the largest, since this also equates to the classifier with the highest level of confidence. [JWHT13]

2.1.4 Unsupervised Learning

Unsupervised learning is another classical category of machine learning, requiring only input data. The main task in unsupervised learning is called clustering and is particularly useful when the correlating outputs, or labels, are not available in the data. The task of clustering is quite similar to classification. In the classification task, data can be seen as already clustered, since it is known which category they belong to and how many categories exists within the given data set. The target of clustering is therefore to obtain this classification for the given data. Since clustering is unsupervised however, one cannot be sure of the accuracy of the obtained model. For example, it is not known beforehand how many clusters the data set should be partitioned into, thus the obtained number of clusters alone could be incorrect. Then there is the issue of incorrectly labelling certain data points, meaning that they belong to the "wrong" cluster for the particular data set one has. Applied to any clustering algorithm is that all observations or data points belong to at least one of the final clusters, while not belonging to more than one of these at the same time.

A visualization of the objective of clustering can be seen in Figure 2.5. [JWHT13]

Since each data point is essentially “forced” into a cluster, another issue one must take into account is outliers. These are data points which do not belong to any cluster, but are instead anomalies from the “regular” data. Finally, validating the obtained clusters is difficult, since there is no way of knowing the actual accuracy of the extrapolated model, as mentioned earlier. In fact, there is no real consensus on which approach is the best for validation. However, the objective for the clustering task is usually chosen to be minimization of the sum of within-cluster variations. The within-cluster variation of a cluster is typically defined as the squared Euclidean distances of the data points from the cluster center. Therefore, one type of validation for the extrapolated model post-clustering could be using this sum of within-cluster variations. [JWHT13]

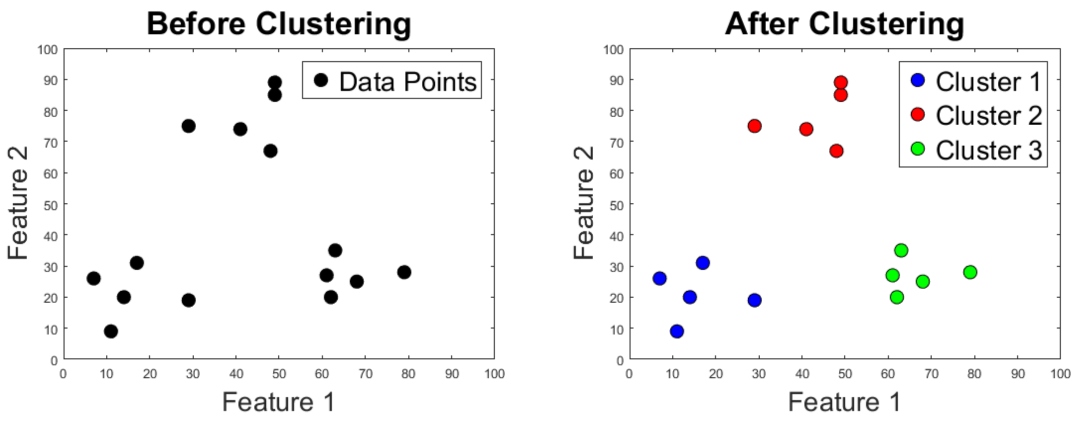


Figure 2.5: Example of the obtained results from clustering. Left panel shows how the input data given is represented before the clustering task. Right panel showcases a clustering algorithm’s grouping of data points.

K-Means Clustering

The algorithm of K-means clustering is a fairly elegant and basic method of unsupervised learning. First, one must state the number of desired clusters, K, that the algorithm should work with. Then the algorithm proceeds with the target of minimizing the sum of within-cluster variations. [JWHT13]

The algorithm initializes by randomly assigning all data points to a cluster, then computing the centers of all K clusters. The next step is to re-assign each data point to the cluster whose center is the closest. After this, one recomputes the cluster centers and does the re-assignment once more. This procedure is repeated until no re-assignment happens. This algorithm always decreases the sum of the within-cluster variations, until it can be decreased no further. [JWHT13]

While the sum of within-cluster variations of the final clusters are always local optimums, there is no guarantee of them also being global. The final result could also differ highly depending on which initial configuration one has. To find the optimal solution, it is recommended that the algorithm is run several times for a given data set, with the clustering that yields the best sum of within-cluster variations selected. While the chance of finding the global optimum increases, this by no means guarantees it. Since the solutions found depend on the initial configuration, it is clear that the algorithm is not particularly robust and quite sensitive to noise. As mentioned previously, it is not clear how many clusters a data set should be partitioned into for an unsupervised task. Therefore, to find the optimal number of clusters, the K-means algorithm should preferably be performed with varying values for K. [JWHT13]

2.1.5 Reinforcement Learning

Reinforcement learning is the deployment of an autonomous agent into an environment, with the task of performing optimally. What optimal behaviour constitutes is defined by the task and the environment itself. An agent interacts with an environment by choosing from a set of actions while in a state and the environment in turn feeds back the agent a new state as well as a reward for a performed action. This cyclical nature gives this category of machine learning its name, “reinforcement”, through the environment reinforcing the agent’s choice of actions. The final objective of the agent is to maximize the cumulative reward across a sequence of actions and feedbacks. [Mit97]

Q-Learning

The most common approach to reinforcement learning, it has two methods of improving its performance. The original version utilizes table look-ups, where the rewards the agent has received is stored in a table. An example of an environment could be a game with a limited set of actions per time step. The agent is then redeployed into the game whenever a game ends, a finite amount of times. For every attempt at the game, each set of actions and corresponding rewards is memorized for future use, in this way the agent explores the strategy space of a game. The actions taken by an agent are typically dictated probabilistically and determined with the help of an equation that is also known as the Q-function.

The Q-function computes the expected reward, Q-value, for any action made by the agent by summing up the potential future rewards. These rewards are usually “discounted” by a factor γ that is dependent on the time step, since long-term future rewards are less valued than short-term ones. The potential future rewards are taken from the table that stored rewards from previous iterations of the game. However, another method that was developed in recent decades. Instead of looking up rewards in a table and then computing the Q-value, a neural network can instead be trained to replace the table, where previously achieved Q-values can be used for training. [Mit97]

2.1.6 Artificial Neural Networks

Artificial Neural Networks (ANNs) is a subfield of machine learning methods that was inspired by the human brain. Attempting to recreate the learning process for neurons in a basic manner, the architecture of the models bear resemblance to the distributed, parallel computing of a neuron. [Mit97]

As the name suggests, a neural network consists of nodes and connecting edges. The edges have associated weights that determine the importance of the particular connection between two nodes. There are several dozens of types of ANNs, with vastly differing architecture, some simplified visual examples can be seen in Figure 2.6 later. There are ANNs for all three classical categories of machine learning methods. One of the oldest ones, the perceptron, is typically used for supervised tasks. An example of unsupervised learning is the Self-Organizing Map (SOM), which handles the clustering task and needs to be fed the number of clusters desired pre-training, much like the K-means algorithm. [Mit97]

Perceptron

Primarily used for supervised learning, the perceptron can do both classification and regression tasks by defining the outputs discretely or continuously, respectively. The architecture has tiers and is linear, going from an input layer to an output layer. In the most basic form, multiple input nodes map to one output node via connecting edges, each with its associated weight. This architecture can vary however, multiple output nodes can be utilized for example. Two additional properties of perceptrons is that the quantity of the nodes stays fixed once selected and that the input layer is typically fully interconnected with the output layer. This means that there are edges with weights connecting every input node to every output node. However, no connections are made between input nodes or output nodes separately. All numerical values in a perceptron are usually normalized so as to balance the range of allowed values in the network. [Mit97]

Signals are aggregated in each node with an activation function. This mathematical processing sums the incoming signals from all connected nodes in the previous layer and provides the perceptron with instructions of how to translate any set of inputs to outputs. Commonly used function are sigmoid or rectifier linear units. The goal of the learning of a perceptron is to select a set of values for all the weights so that error it makes when translating a set of inputs to outputs is minimized. By far the most popular choice of method for training a perceptron is the backpropagation algorithm, which iteratively updates the values of the weights of the network. The process of training is therefore a gradient descent, attempting as much as possible to move in directions where the function space slopes “downwards”. The function in question here is a cost function, defined by the aforementioned error measure in some way. Although the target is to find the global minima, it is difficult to avoid landing in local minima. To avert this, stochastic gradient descent can be used or momentum can be introduced to the regular gradient descent. [Mit97]

The architecture of the perceptron can be extended to incorporate layers additional to the input and output ones. These are called hidden layers and are placed in between the inputs and outputs, to create a more complex network structure that can make more advanced predictions. This extension of the perceptron is called the Multi-Layered Perceptron (MLP). The number of hidden nodes per hidden layer can also vary, while the layers typically maintain full interconnections as in the case with the basic perceptron. The number of hidden nodes should not be excessively large, as this leads to overfitting the MLP to the data, nor should it be chosen to be to small, as the complex relation in the data might not be mapped properly. [Mit97]

2.1.7 Deep Learning

In recent years, deep learning has grown as a subfield of machine learning that has steadily become more popular. This is due to its ability to solve more complex problems, such as speech recognition and computer vision. A popular example of deep learning implementation is the AlphaGo program developed by Google, which defeated the European champion of Go, Fan Hui, by $5-0$ in October of 2015 [SHM+16]. Deep learning is normally defined as cascading layers of non-linear machine learning algorithms, which enables the learning of highly complex patterns simple or single methods would not be able to achieve normally. [DY14]

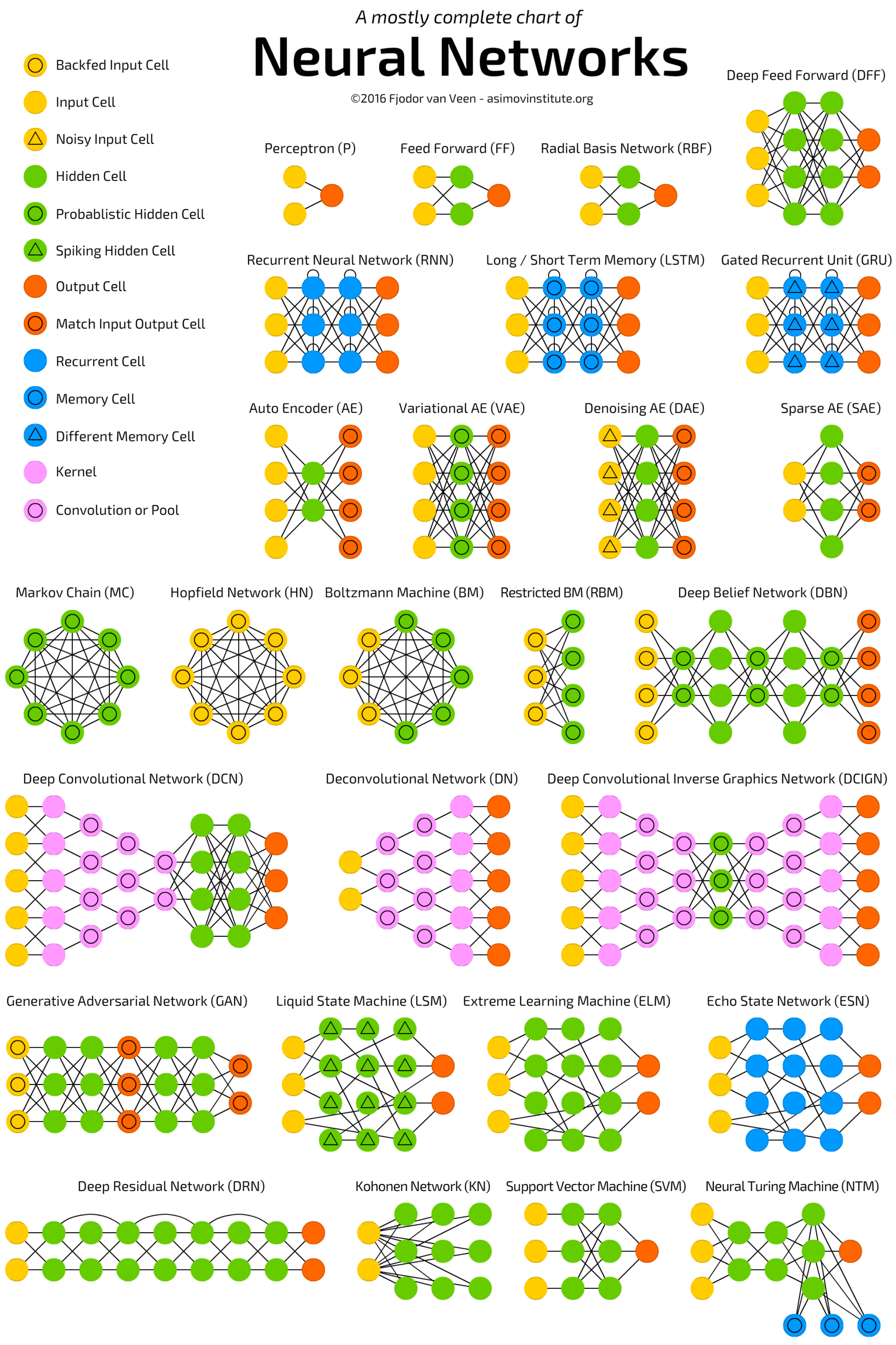

A mostly complete chart of
Neural Networks
©2016 Fjodor van Veen - asimovinstitute.org
Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Kohonen Network (KN)
Support Vector Machine (SVM)
Neural Turing Machine (NTM)

Figure 2.6: The architecture of different types of ANNs. Picture taken 2016-03-01 from `http://www.asimovinstitute.org/wp-content/uploads/2016/09/neuralnetworks.png`.

2.2 The Evolved Packet Core Network

The Evolved Packet Core Network (EPC), is currently the largest and most complex network around the world, whose main responsibility is the management of wireless communication between various devices across the world. [Fir]

The architecture of the EPC network, which is shown in Figure 2.7, was designed by a group of seven telecommunications standard development organizations known as the 3rd Generation Partnership Project (3GPP) which serves as a provider of a standardized environment to produce new technology within the telecommunications industry. [3GP]

The network provides support for two types of wireless communication technologies, the Long Term Evolution (LTE) as well as Wi-Fi communication. [Fir]

2.2.1 Architecture

The EPC is purposed with a “flat” architecture that handles the payloading (transportation of data traffic) more efficiently, seen from both cost as well as performance perspectives. The architecture of the network is divided into two planes, which can be seen as the two phases of any wireless communication: [Fir]

- **Control Plane:** also known as the c-plane, which is responsible for registering and connecting newly started devices into the network. [Fir]
- **User Plane:** also known as the u-plane, which is responsible for handling the transportation of data from the user’s device to external networks such as the Internet. [Fir]

When a subscriber turns on their wireless communication device, known as a User Equipment (UE), the device sends an activation signal to the nearest local radio base station, known as eNodeB (eNB). The eNB then transmits the signal to the EPC network in order to register the user into the network. This is done via the signalling processes of the c-plane, which verifies that the subscriber is legitimate, connects said subscriber to the network and starts to monitor the subscriber’s position and status. [Fir]

After the subscriber has been verified and granted access to the network, the subscriber is then able to communicate over the network by sending and receiving data. This data is transmitted over the signalling processes of the u-plane, which handles the sending and receiving of data from the subscriber to external networks. An example of an external network is the Internet. [Fir]

The EPC network is made up of numerous components, which are known as nodes. The next section provides general information on those nodes and their responsibilities. [Fir]

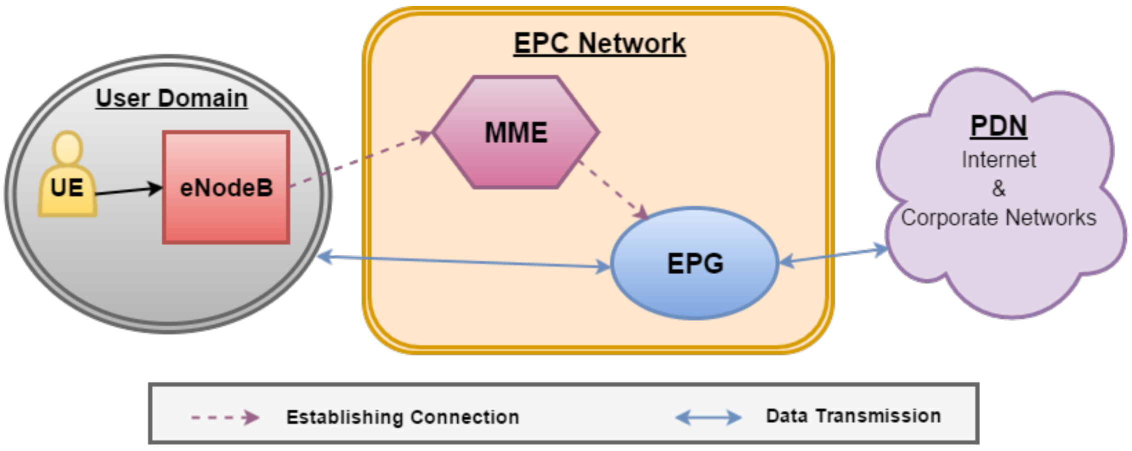


Figure 2.7: An Overview of the Evolved Packet Core Network.

2.2.2 Main Nodes

Each of the two planes of the EPC network, can be viewed as a centralized network that is made up of a main node which is connected to a number of supporting nodes. [Fir]

Mobile Management Entity

The Mobile Management Entity (MME) is the main node of the c-plane. It is the node to which every subscriber must connect and register in order to be able to get access to the network. The node handles many activities, such as first time registry of new subscribers, authentication and verification of existing subscribers as well as keeping track of the subscriber's positions and status. [Fir]

Evolved Packet Gateway

The Evolved Packet Gateway (EPG) is the main node of the u-plane. It is the node through which all of the subscriber's communications goes through, which includes making phone calls and having access to the Internet. Its main responsibility is sending and receiving the subscriber's data from the UE to external networks, such as the Internet, and vice-versa.

The node is divided into two sub-components. The first one is the Serving Gateway (SGW), which handles sending and receiving of data from the UE into the EPC network, and the second is the PDN Gateway (PGW), which handles sending and receiving of data from the EPC network into external networks.

2.2.3 Support Nodes and Interfaces

Each of the two main nodes of the EPC network are connected to a number of supporting nodes which handle various activities and features that assist the main node in achieving its tasks.

The main node of the c-plane, the MME, is connected to the following supporting nodes:

- **GMLC:** or Gateway Mobile Location Centre, is responsible for tracking the subscriber's location.
- **HSS:** or Home Subscriber Server, is a form of a database that keep track of each subscriber's information.
- **EIR:** or Equipment Identity Register, is responsible for checking the UE's legitimacy, whether the subscriber's bills are paid, or whether the mobile device has been stolen and blacklisted.
- **E-SMLC:** or Evolved Serving Mobile Location Center, which calculates the velocity and final position of a UE.
- **CBC:** or Cell Broadcast Center, responsible for the management of warning messages sent through the Public Warning System (PWS).

The main node of the u-plane, the EPG, is connected to the following supporting nodes:

- **PCRF:** or Policy and Charging Rules Function, is responsible for setting up rules and polices on each subscriber, such as parental control.
- **OCS:** or Online Charging System, as the name indicates, the node is responsible for handling all charges of services done online.
- **CDF and CGF:** or Charging Data Function and Charging Gateway Function, responsible for creating records for the calls that subscribers make in order to charge them for it later. These records are commonly known as Calling Data Records (CDRs).
- **BS:** or Billing System, responsible for charging the subscribers with bills of the services they use.

Each of the support nodes communicate with its respectful main node via real-time communication interfaces. Those are specific protocols that are allow the transfer of certain types of data between the main and support node. Due to the abstraction of the interfaces, and the fact that it might reveal sensitive information about the network, the interfaces will not be covered in any more details within this thesis.

2.2.4 Network Data

The EPC network uses plenty of monitoring methods in order to keep track of everything that is happening within its main and support nodes. These methods result in vast amounts of data that are stored in log files and databases for future access. Below, a few of the parameters are introduced within these log files, which are relevant to the results of this thesis.

- **Counters and Gauges:** Counters are positive integer parameters that are used for monitoring performance during the execution of a task within the network. There are plenty of counters that are keeping track of the status of the node within the network, such as CPU and memory usage, number of users in the network, number of calls a user makes per hours and so on. Counters are incremented per every occurrence that is related to the counter within the execution of a task. Counters have a maximum threshold, and they are reset when they reach that threshold. Gauges are also positive integer parameters that work similarly to counters, except that while the value of a counter can only be increased, the gauges can be both increased and decreased. Thus, the measurements of gauges are taken as a delta over time.

- **Events:** The EPC network also contains *Event Based Monitors*, which are used for registering information about the behavior of the nodes during the execution of a task. Thus, an event indicate a certain activity that occurred during the execution of a task. Examples include whenever a user connects or disconnects from the network, sends a message or makes a call. Each event contains information such as the time when the even took place, how long the event was and the status of the event (whether it succeeded or not).
- **Signals:** Signals within the EPC network involve all types of communication that occur internally between the nodes of the network, as well as communication between the network and external sources. The main type of signals that are relevant to this thesis are network packets and communication protocols which can be captured using any traffic monitoring tool.
- **Alarms:** All the parameters of the EPC network are monitored using automated processes which are known as *supervisors*. If any of the parameters exceeds a certain threshold that it is not supposed to exceed or can results in a danger to the network, alarms are triggered that notify the administrators of the nodes about the possible danger. Some alarms are automated in the sense that when the alarm is raised, a certain action will be taken automatically. However, most alarms are monitored and addressed manually.

Chapter 4 discusses the use of the above parameters, as part of the implementation of the machine learning algorithms in more details.

3
Approach

The research and thesis structure is divided into several phases, all of which are described in the following sections.

3.1 Research Methodology

The most suitable choice for conducting the research for this thesis is a research methodology known as design research [CJB04], where the goal is to create a service, feature or a full product.

This is due to the fact that the research includes a combination of extensive literature review, innovating ideas of integrating machine learning into EPC and details of the design, implementation as well as expected results of those ideas. It is expected that prototypes should be able to be created as proof of concepts for all of the ideas.

This chain of research indicates that a design research approach is the most suitable type of methodology for this paper. Action research [ALMN99] might be a useful alternative, however most theses that use this type of research have a tendency of being done over a longer period of time. A design research is preferable since the outcome of this paper is intended to include tangible results that the company might be able to use in the future.

3.2 Research Questions

The thesis seeks to establish answers to the following questions:

- ***RQ1:*** What kind of machine learning algorithms have been implemented within the telecommunication industry, particularly for the EPC network?
- ***RQ2:*** How can improvements be made to the current form of the EPC network using machine learning?

In order to find an answer to the first research question, an extensive phase of literature review is needed, where the results are summarized and presented in a structured manner to both the company and the university. As for the second question, the authors intend on studying the various nodes of the network, meeting experts of said nodes, and work together in order to come up with ideas where machine learning can be used within the EPC network.

3.3 Research Objectives

The purpose of the thesis is to identify areas within the EPC network with potential for improvement through the utilization of machine learning algorithms. This goal is accomplished by fulfilling the following objectives:

- Studying research papers on existing implementations of machine learning within the telecommunication industry, and evaluating their applicability within the EPC network.
- Studying the EPC network and its various components (nodes), to understand what tasks they perform and the type of data they work with.
- Interviewing experts on the main and support nodes of the EPC network to further understand the nodes' functions.
- Establishing ideas where machine learning algorithms can be utilized within the EPC network in order to facilitate some of its tasks.
- Providing detailed description of the implementation of each idea, including the machine learning algorithm to be used and the type of data to work with as well as expected results and benefits of the idea.
- Recommending the best or most suitable ideas that can be prototyped and presented to the company.

Depending on the time frame of the thesis, prototyping one or two of the proposed ideas as proof of concept could be possible. In the case where a prototyping phase is not possible due to lack of time or resources, it can be performed as the next step of the study outside of the scope of the thesis.

3.4 Research Approach

The research is divided into a number of phases that can be seen in Figure 3.1. The *planning phase*, involves all the preparations needed for the thesis. The phase involves writing a thesis proposal that contains an accurate formulation of the problem and research questions to be answered. This is followed by a planning report that depicts the objectives and milestones of the thesis. The planning phase also includes studying related work in order to establish the research's gap, as well as preparations for the study and interviews.

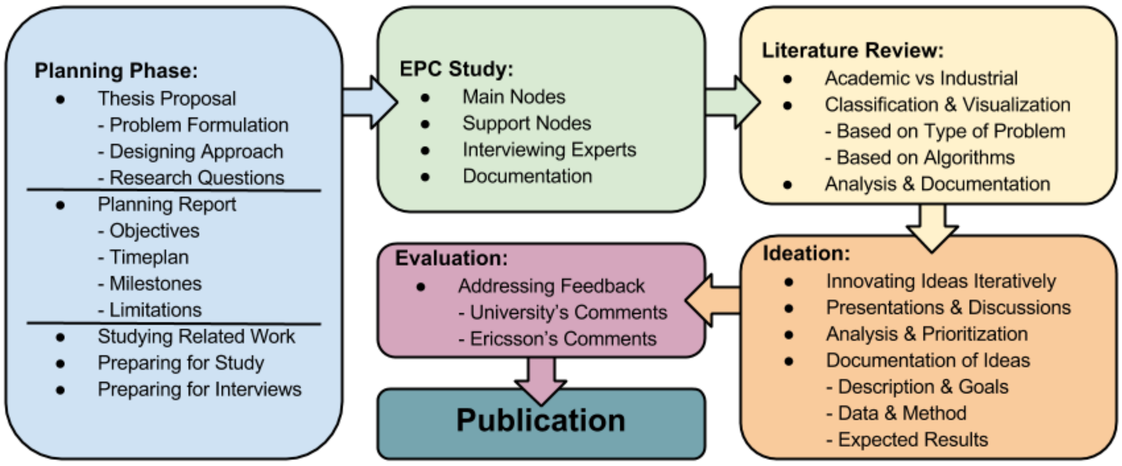


Figure 3.1: Research Approach.

The next phases involves a study of the Evolved Packet Core network, which is described in Section 3.5. The literature review phase is described in Section 3.6 and the process of innovating ideas is detailed in Section 3.7. The final stage of the thesis involves the evaluation and defense of the thesis, where all findings and results of the thesis is presented. At this stage, *feedback* is given from both the university's examiner as well as Ericsson, thus the phase involves addressing all given feedback and submitting a thesis paper that is ready for *publication*.

3.5 Study of the EPC Network

An important aspect of this thesis is the understanding of the Evolved Packet Core network. This is because in order to be able to integrate machine learning algorithms into any of the system's nodes, an extensive comprehension of the nodes that make up the network along with the various tasks they do and the type of data they work with, is needed. However, the EPC network is one the most complex networks that exist today and understanding its various components is a process that takes both a significant amount of time and effort. It is thus important to establish a strategy by which it is possible to get a high level understanding of the system's nodes, including their tasks and data, during a relatively short period of the thesis.

3.5.1 Divide and Conquer

In a case similar to that of solving big and complex problems, the approach taken to the study of the EPC network is by diving it into smaller parts that are easier to understand. Luckily, the network itself is made up of two planes, the control level and user level, each with its own hierarchy of main node and support nodes. Thus, starting with a study of the main nodes of the system and following with each of the supporting nodes, for each plane, seems appropriate.

3.5.2 Reading Specifications

One of the benefits of working on systems that belong to large companies such as Ericsson's is that the majority of their systems, both hardware and software, are documented in great detail. Each of the main and support nodes of the EPC network has its own document that details the functionality, features and data that the node has. Thus, a great deal of information on the various nodes of the systems can already be found in a structured manner, which decreases the complexity of the study.

3.5.3 Conducting Interviews

While the documentation of the various nodes is useful to understanding the functionality of each node and the type of data it works with, the documents themselves are limited. This is where the other part of the study comes into play, which is conducting interviews with developers and designers of the various nodes of the EPC network. These are experts in the systems that they work with, whom are able to give a quick yet detailed overview of a node's functionality and the data it works with.

Due to time constraints, including the busy schedules of these experts, the number of people that can be interviewed and the number of times an interview can be conducted will be limited for the duration of any thesis. However, there are certain rules that can help in gathering as much information as possible [Row12]. These rules include:

- Preparing all relevant questions before the interview. A pre-study of the component is helpful in brainstorming such questions.
- Presenting a brief overview of machine learning and explaining the purpose of the interview in order to give the interviewee an understanding of the research questions to be answered.
- Ensuring that all questions are open ended, in order to allow for extended discussions that could lead to an extended discussion about the component.
- Ensuring documentation of all answers.

In normal circumstances a list of questions should be appended to the thesis [Row12]. However, since some questions tend to touch upon sensitive information that belong to Ericsson, no list is presented as part of this thesis.

During these interviews, engaging the interviewees in discussions about how a node can be improved using machine learning is vital. The knowledge that the expert has on the node makes him or her an ideal candidate for such type of discussions, which can assist in coming up with ideas on the integration of machine learning algorithms within that particular node.

3.6 Literature Review Process

This section details the process by which the literature review is carried out in order to provide an answer to the first research question.

3.6.1 Data Sources

In order to find as many primary studies that are related to machine learning within telecommunications as possible, there needs to be a number of sources where research papers that are related to this subject can be gathered [BPS12]. The main data sources that are used for the search include the libraries of *ACM*, *ArXiv*, *Digital Library*, *CiteSeerX*, *IEEE Xplore*, *Inspec* & *Springer*. These libraries maintain some of the oldest as well as some of the latest published research papers, and their contents are regularly maintained. All of the aforementioned libraries are accessible online via *Google Scholar* and other search engines.

3.6.2 Search Strategy

The benefit of using online search engines such as *Google Scholar* to access the aforementioned libraries is that quick as well as customized search queries based on keywords that are related to the researched subject can easily be done. However, even with the proper keywords these search engines returns thousands of papers. Thus it is necessary to create specific search queries that filter out publications that are not relevant to the thesis [BPS12].

The search strings reflect the aim of the thesis, that is machine learning within the telecommunication industry. It is also important to use proper Boolean expressions such as *AND* as well as *OR* to concatenate those strings, which helps in filtering out irrelevant publications. The formulated search strings, deemed relevant and yielding useful results, include the following phrases:

- Machine Learning OR Data Mining AND Telecommunication.
- Machine Learning OR Data Mining AND Evolved Packet Core.
- Machine Learning OR Data Mining AND 4G.
- Machine Learning OR Data Mining AND LTE.
- Machine Learning OR Data Mining AND WiFi.
- Machine Learning OR Data Mining AND MME.
- Machine Learning OR Data Mining AND EPG.
- Machine Learning OR Data Mining AND Ericsson.

3.6.3 Criteria of Inclusion

In order to further limit the number of papers and articles that are found using the search strings discussed in the previous section, a number of inclusion criteria must be established. These criteria help exclude papers that are irrelevant to the study.

Based on [BPS12], there are several standard criteria of paper inclusion, the paper must:

- be written in English,
- be published at peer reviewed conferences and journals,
- provide solid references for the mentioned facts,
- not be older than a certain publication period, such as the 1990s.

However, the following additional constraints are imposed. The paper must:

- be explicit in the problem it is attempting to solve using machine learning,
- solve a problem that is directly related to GSM, 3G or 4G communication standards,
- specify the algorithm itself or the type that was used to solve the problem,
- specify the type of data that is used for training, validation and testing,
- state the extrapolated or expected results of the algorithm.

In the cases where a paper fails at most one of these inclusion criteria, the paper is deemed acceptable for analysis, otherwise it is omitted.

3.6.4 Data Extraction and Analysis

The next step involves the extraction of relevant and useful information from the papers that are found through the search. The papers are then analyzed and summarized as the results of the literature review [BPS12]. Thus for each paper found, a data extraction process that works as follows is deployed:

- Reading the paper's abstract and introduction in order to get an overview of the paper and ensure relevance to the study.
- Establishing the problem that is being solved using machine learning.
- Identifying which machine learning algorithm is used to solve the problem.
- Establishing how the algorithm was implemented in order to solve the problem.
- Identifying what kind of data is being used for the training, validation and testing phases of the algorithm.
- Identifying the results of testing the algorithm.
- Understanding the benefits of deploying the solution to the particular problem.

As part of the analysis of the paper, an explanation of how the particular solution is applicable and can be implemented within the EPC network is included. Each paper is summarized and documented as part of the results of the literature review, where the summary contains a:

- summarized description of the paper and its aim,
- abstraction of the method that was used in order to solve the particular problem that is being addressed in the paper,
- description of the results of the implementation,
- discussion around the applicability of the solution within the EPC network.

The various papers found are categorized into generic groups that are representative for the type of problem that is being solved within the telecommunication industry. Each paper is also classified by the type of algorithm that is being used in order to solve the problem. This is done through a visualization of the types of problems and types of solutions that are found through the literature review. This serves the purpose of making it easier to follow up on the literature review results and further helps readers that are interested in certain types of problems to quickly find what the desired solution.

3.7 Innovation of Ideas

This section details the process that was used for innovating solutions and ideas that involve the integration of machine learning algorithms within the EPC network.

3.7.1 Iterative Approach

Attempting to produce viable ideas that involve the use of machine learning for a network as complex as the EPC can be overwhelming. Obstacles that can appear along the process includes losing sight of the original objective, coming up short on ideas or having an unbalanced amount of ideas for different nodes in the network. It is therefore necessary to approach this phase with a method that allows for coverage of all the nodes in the EPC network, including the possibility of going back to any node at later steps and attempt to innovative additional ideas to it.

The approach to this problem implemented in this thesis is similar to the iterative software development processes. The idea is to spend certain amount of time on each of the main and support nodes of the system and attempt to think about the node from two perspectives. The first is internal, where the goal is to achieve some internal function of the node using machine learning methods. The second perspective is external, which means viewing the role of the node from the network's perspective and achieving some function here. The function achieved can be either complimenting, meaning it adds some feature to the node, or substituting, meaning it replaces an existing function in exchange for improved performance in the node.

This process is repeated for each node within the system at least once, depending on the time that is spent on the node and the time constraints of the thesis. The benefit of going over a node more than once is that the likelihood of coming up with more ideas for the particular node at later step increases, particularly when working with other nodes of the system.

3.7.2 Conducting Presentations

One of the reasons behind Ericsson's interest in the thesis is because the company is attempting to incorporate machine learning as a new standard of development within the telecommunication industry. The company's goal is to spread awareness of the possibilities that could be achieved throughout the use machine learning amongst its departments. Thus, an important part of this thesis is the introduction of the concept of machine learning to the employees. This involves conducting presentations with various teams from different departments of Ericsson, all of which are involved in the research and development of the EPC network.

This is particularly useful for the thesis, because this type of presentations cultivates discussions among the designers, developers, testers and managers of the various nodes of the network, on how machine learning could be used within the EPC network. Throughout these discussions, the attendees will be able to come up with their own ideas or solutions to address problems or tasks they have experience from.

Thus, the role of the presenters in said presentations is as to:

- present the concept of machine learning from a high level perspective without going into complex details,
- differentiate between standard methods of programming and machine learning so that the attendees can get a better picture of how machine learning works,
- present exciting examples of what machine learning is being used for today to increase the motivation of the attendees,
- introduce what work has been within the telecommunication industry with regards to machine learning,
- present ideas of machine learning uses within certain nodes of the EPC network that the authors have produced, which the attendees can find relatable,
- motivate the attendees to open discussions on the problems that they work with and how machine learning can be used to help making their work easier,
- take note of any ideas that may result out of those presentations, for further analysis and documenting.

The number of attendees per presentation is expected to be between $10 - 15$ people, ideally allowing for open discussions of enough depth while giving everyone a chance to talk. Due to the usefulness of the presentations, to both the company and the thesis, conducting as many of these as possible within the time frame of the thesis is optimal and prioritized.

3.7.3 Analysis of Results

Due to the enormity and complexity of the EPC network, the expectation is to end up with a preliminary list of around 30 ideas and solutions. This sizable number necessitates further analysis of each of the proposed ideas in order to assess their validity as well as importance to the EPC network. The analysis includes categorizing the ideas or solutions into generic groups, much like the process in the literature study phase described in Section 3.6.4. These groups are based on the type of problems that they attempt to solve as well as the type of machine learning algorithm that is recommended to be used.

The target of the analysis is to establish a final list of handful of ideas that can be recommended to Ericsson for immediate prototyping. This is done by consulting with technical supervisors at the company, in order to establish the viability of each idea within the EPC network. This includes determining the level of its complexity, time and effort needed to prototype it as well as its benefits to the company.

The priority is to establish as many ideas as possible. However, if there is enough time left in the thesis, prototyping one or two of the innovated ideas as proof of concepts should become a priority.

3.7.4 Presentation of Results

The results of this phase is primarily presented in two parts. The first part involves detailed documentation of each idea or solution that is produced throughout this phase. It serves as the template that needs to be followed when prototyping a particular idea. Thus, for each idea, the following is presented:

- A suitable title for the idea.
- A description of the idea, which includes listing the problem that is going to be solved or the task that is going to be automated, using machine learning.
- Detailing the implementation of the idea. This includes the node within the EPC network, the type of data to be worked with, the machine learning algorithm to be used, the method or process that needs to be followed and the type of tests that needs to be applied.
- An overview of the idea's result.
- A description of the benefits that are gained by employing the idea.

Some underlying information might be omitted from the results, in the cases where it might reveal sensitive information that belong to Ericsson. In such cases, the omission of such information is noted in the corresponding part of the idea.

The second part of the results includes the visualization of the ideas into their respective categories based on the results of the analysis, similarly to Section 3.6.4. This serves the purpose of making it easier for readers that are interested in specific types of problems or algorithms to quickly find what they are looking for.

4
Results

The results come in the structure and order described in Chapter 3. Firstly, the results yielded from the literature study phase are presented in Section 4.1. Secondly, Section 4.2 contains a list of the ideas we innovated.

4.1 Prior Art

After an extensive literature study, previous machine learning achievements made within the telecommunications field are presented according to application areas. Six areas were identified where machine learning had been implemented. Each area contains several methods for solving specific problems, summing up to a total of 16 research papers found. A tree graph of the found research papers is presented in Figure 4.1. In the figure, the squared, magenta boxes represent categories of applications within the telecommunications industry. Each of these boxes branches further out to multiple circular boxes of varying colors. These represent individual research papers that are detailed in later sections, with the coloring in the figure highlighting what type of machine learning method has been applied.

The areas of application, within telecommunications, for machine learning are Internet traffic classification, performance of the network and fault detection in the systems. The rest of the categories are directed more towards subscriber or user profiling. These included detecting fraudulent customers, churning subscribers and predicting the users' geographical trajectories.

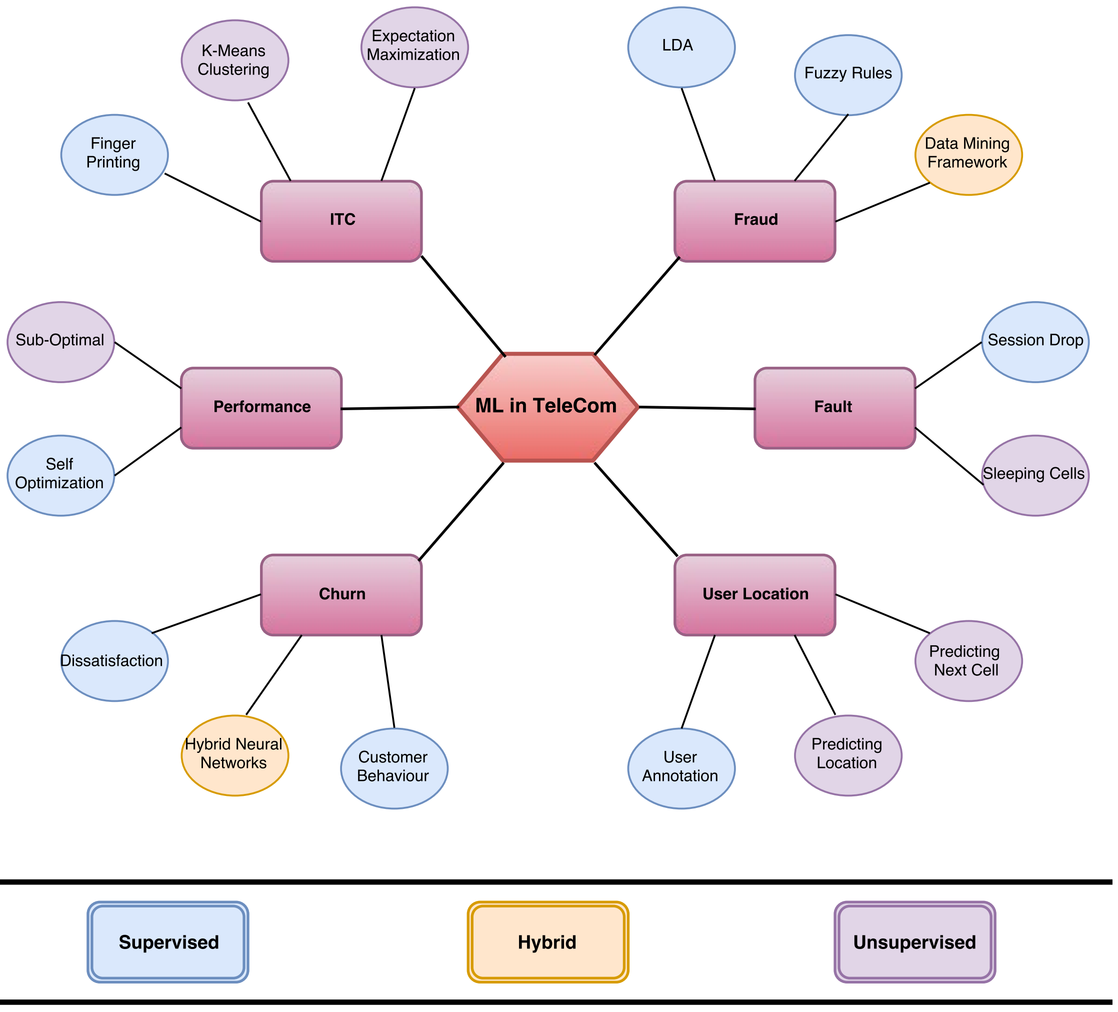


Figure 4.1: A tree graph structuring up the research articles according to their usage of machine learning algorithm and application area. In total, six types of applications were found from the literature study (categories marked in magenta colored boxes).

4.1.1 Internet Traffic Classification

Internet Traffic Classification (ITC) is the subject of identifying types of data payloaded through a network. An example is protocols of Internet data.

Classification of IP Traffic Using Statistical Fingerprinting [CDGS07]

Description: This method involves the identification and classification of individual protocols from captured IP traffic.

Method: The authors begin by identifying a set of properties which are, (packet length, inter-arrival time, packet arrival order). They are combined together into a structure which can be used to uniquely identify the captured IP packets, which was labelled as a "protocol fingerprint". The authors then used training and classification based on normalized thresholds for classifying the packets. The training set's data are then analyzed to build the protocol fingerprints as a vector of probability density functions (PDF), where each function represents the protocol fingerprint values. Packets are then matched against the PDFs in order to establish the most statistically compatible PDF that describe that packet and scored are given to indicate the distant of a packet from a given PDF. The higher that score is, the more likely that the packet is generated by the protocol that is described by that PDF.

Results: The algorithm seems to have a high accuracy, over 90%, in classifying a number of protocols including HTTP and POP3. The algorithm is able to classify the protocols using a few packets of the traffic's flow. However the algorithm suffers from the assumption that the captured packets always include the full trace as well as the first packet of each flow.

Application in EPC: There is a large number of properties that are provided by tools such as Wireshark which can be used in order to create a similar fingerprint to the one used by the authors. Thus implementation of this algorithm is quite feasible within the EPC network for classifying the various IP traffic.

Classification of IP Traffic Using the Expectation Maximization Algorithm [DLR77]

Description: The goal is to identify the Internet protocols (HTTP, FTP, SMTP, IMAP, NTP and DNS) from a six hour capture of a full flow packet trace.

Method: The classification is done using an unsupervised clustering algorithm, which is the Expectation Maximization Algorithm. The algorithm groups the captured packets into a small number of clusters, from which further classification rules can be created. Next, using the classification rules, packets that do not comply with those rules are removed, and the resulting packets are reused in the algorithm.

Results: The algorithm was successful in identifying rules and features that could be used to identify groups of the aforementioned Internet protocols, however it does not identify individual protocols accurately.

Application in EPC: Given the large traffic and the amount of Internet protocols that exist within the EPC network, it is possible to use this algorithm as a first step in grouping and identifying protocols that share similar characteristics. This can be done by capturing a full flow trace using Wireshark, and then applying the EM algorithm as explained in the referenced paper to group the IP traffic.

Classification of TCP Traffic Using K-Means Clustering [BTA+06]

Description: The goal is the early identification and classification of the various types of TCP protocols in a traffic flow, using only a small portion from the beginning of the traffic.

Method: Using a simplified version of the K-means clustering method, where it is used as an unsupervised machine learning. The algorithm groups the traffic flows into clusters using their first P packets as input values, thus resulting in a P-dimensional space, where the coordinates are represented in the size of the packet. The algorithm then clusters the packets based on the Euclidean distances between the coordinate, where each cluster holds the minimum distance between the packets.

Results: The algorithm was tested on a one hour trace of a TCP flow, and it managed to correctly cluster and identify the inputted traffic, using only the first five packets of the flow. It was able to identify individual protocols such as POP3 and SMTP with over 80%. The method however, assumes that the first few packets of the trace contains a distinct sequence of messages which can be used for classification. It also requires that the input is a trace of TCP flows, thus including non-TCP flows could impact the results negatively. Finally it assumes that the captured trace always contains the start of each flow fully.

Application in EPC: Based on the conditions above, it is important to note that this algorithm will work as a second stage classification. In the first stage, the TCP traffic can be grouped out in an alternative algorithm such as the EM algorithm, and then using this algorithm, it is possible to classify the traffic into individual protocols with a relatively good accuracy.

4.1.2 Churns

Churns is a term defined as users unsubscribing from services. When applied within the telecommunications industry it refers to customers that discontinue their usage of a telecommunications service provided by a specific operator, usually meaning customers switching from one operator to another. Churns can be divided into two categories, voluntary and non-voluntary. Non-voluntary churns are subscribers whom had their subscriptions forcefully cancelled by the operator, usually due to failure to pay the bills. The main focus of the telecommunications industry is however on voluntary churns, which is what most of the following papers have examined.

Customer churn prediction by hybrid neural networks [TL09]

Description: In this article, two hybrid models of supervised and semi-supervised learning were implemented in order to predict churns from a data set provided American operator. These were then compared with a simpler supervised learning model to show that the increased complexity of the hybrid models could generate better results.

Method: Utilizing a data set comprising of $51,000$ subscribers, with each subscriber being customers with their respective operator for at least the past six months, the hybrid models were tasked to predict churn in the second month after the time window of the data set. The supervised learning model cascaded two ANNs that performed data reduction followed by churn prediction while the semi-supervised learning model used a SOM to perform the data reduction, followed by an ANN to predict the churn. The data reduction phase filtered out and discarded data points that were difficult to classify, also known as outliers, since the prediction models often yielded incorrect results for these. The training was then performed on the remaining data points. A single ANN trained on the entire original data set was used as the baseline model to compare performance with. Three data sets were then tested against, one general and two fuzzy. The fuzzy ones comprised only of the outliers from the data reduction yielded by the ANN and SOM, respectively.

Results: The baseline model was found to have more than 88% in prediction accuracy on average. The two hybrid models were found to outperform the baseline model on the general testing set. Across the fuzzy testing sets, the ANN+ANN hybrid model was found to be the best performing model, with the SOM+ANN hybrid model underperforming even when compared to the baseline model. In conclusion, the ANN+ANN hybrid model was the most stable one and performed better than both the baseline and the other hybrid model, SOM+ANN.

Application in EPC: These results show that an ANN+ANN hybrid model works excellently as a churn predictor and indicates that the relations or patterns found in the data of churning customers are more complex and require more sophisticated models in order to be predicted better. Since the MME node already handles substantial subscriber information, this implementation can be replicated in the EPC network directly.

Predicting Customer Behavior in Telecommunications [YWD04]

Description: This paper examines supervised learning that predicts the likelihood customer churns for the coming time period and identify the reason of churning. Findings were somewhat inconclusive however, with the main lesson being that the storing and handling of data is crucial if machine learning of any type is to be implemented.

Method: The research done by the authors in the article is split into two parts. Described first is the data preparation for processing by an ensemble of classifiers and then the actual application of an ensemble algorithm. The customer churns data sets were considered inconsistent, usually being both non-stationary in time and sparse or incomplete across parameter space, that is occasionally under-representing certain parameters. The model implemented is one that takes historical data from a certain chosen time-interval and that predicts probability of customer churn for an equivalent coming time-interval. When pre-processing data, the authors utilized domain knowledge for feature selection. To overcome the issue of dependent parameters, they introduced extra parameters to link other parameters. Though the details of the model are not specified explicitly, the ensemble of classifiers are implied to attempt both MLP and Naive Bayes variants, with periodic updates to the training made in order to keep the model fresh and relevant.

Results: After adjusting the imbalanced data sets, the results with the ensemble of classifiers were found outperform individual classifiers. The authors also noted that the training data could be augmented with spontaneous accepters, that is subscribers whom previously accepted a marketed offer and joined the operator, as well as separating between voluntary and non-voluntary churns. In conclusion, they found that there was a need to improve the storing of data, since the achievable preparations had limitations to how much they could improve the data sets.

Application in EPC: From the paper, it is clear that the initial storing of data is vital in order to be able to get something meaningful out of it. This paper serves as a lesson that parsing of data must be a priority, since the data could serve purpose for future analysis.

Predicting Subscriber Dissatisfaction and Improving Retention in the Wireless Telecommunications Industry [MWG$^+$00]

Description: This article examines the use of several supervised learning methods in order to predict churn, while also attempting to determine the underlying reason for churning. The purpose also includes finding what type of incentive should be offered to the subscribers in order to retain them.

Method: The authors utilized three machine learning methods. These were logit regression, C5.0 decision tree and a neural net consisting of single hidden layer ANNs with weight decay. These were then boosted with AdaBoost and compared with the non-boosted variant. The data the authors had access to were provided by one undisclosed operator and contained $47,000$ subscribers in one region in the US. The data included the usage, billing, services and credit amongst others. An important process of feature selection was also using domain experts, with this aspect included the authors had representations for the data. One naive, consisting of a 148-element vector as inputs, and one sophisticated variant, made up of a 73-element vector as inputs.

Results: The results were divided into three parts. Analyzed first was the predictive power of the methods. The authors found that utilizing the sophisticated data representation yielded the most optimal lift curve, with the boosted neural nets strongest performing method as well. Secondly, analysis of decision making when offering incentive was shown. When applying a likely scenario, where all subscribers could not be contacted and all that were offered incentives could not be retained, boosted neural nets were again found to perform the strongest, with logit regression coming in not too far behind. Finally, the authors applied their predictors to live real-world data in the form of a treatment and a control group. Up until this point, the predictors had been trained and tested against data from the same time periods. For the real-world evaluation, the test window was simply shifted from the training window. The findings were that when the operator attempted to incentivize potential churners in the treatment group, a 40% drop in churn was achieved in comparison to the control group, with remarkable savings in cost for the operator accomplished.

Application in EPC: Since the EPC networks already handle all the information the authors were provided with by the undisclosed operator, this application can be replicated by the TSPs directly. Although of most interest to the operators, the TSPs can also implement this technique and then package this as an additional service to operators in order to expand existing business deals.

4.1.3 Frauds

Fraudulent behavior exhibits itself in telecommunication networks as subscribers that uses services with no intention of paying the expected charges of said services. This commonly happens through temporarily existing subscribers, customers misusing their subscription terms and users whom superimpose upon (or hijack) another subscriber profile, thus leaving this subscriber with the bill to pay.

A data mining framework for detecting subscription fraud in telecommunication [FS11]

Description: A hybrid, semi-supervised approach for identifying subscription frauds is employed with the purpose of detecting patterns of residential and commercial subscribers' behavior. These patterns can then be used to differentiate between fraudulent from non-fraudulent customers within the residential subscription realm, since residential subscription is often offered at a lower price than commercial ones.

Method: The eight, two-month periods worth of data provided to the authors were of $25,000$ subscribers in Tehran and included detailed calling and billing records along with sensitive customer information. First phase of the model was pre-processing, which included dimensionality reduction via PCA of the data along with discarding of incomplete subsets of the data. Second phase of the hybrid model utilized unsupervised learning for cluster analysis and removal of groups of potential outliers. The methods employed in this phase were SOM, k-means and a hybrid of both. The third and final phase of the model included supervised learning for the classification of the clusters. These algorithms included C4.5 decision trees (with and without pruning), ANNs (trained using back-propagation and conjugate gradient descent while varying parameter settings), SVMs and ensemble methods. The ensembles included bagging, boosting (with AdaBoost.M1), stacked generalization and voting (consensus and majority).

Results: The pre-processing phase lowered the number of features from 56 to 25, with the clustering analysis taking the number of input features to 28. The ensemble classification methods were found to be more robust than single classification methods, where the best performing single method was the SVM with an accuracy of 88.2%. Amongst the ensembles the boosted trees were found to be the best performing method overall with an accuracy of 93%. The boosted trees falsely classified 8.63% of the commercial and 6.15% of the residential customers, respectively.

Application in EPC: With the added component of a final check-up from the operators on the accused customers, said operators could stand to reduce their own costs and losses through fraudulent behavior from residential customers without significantly harming customer relations. This shows the fiscal usefulness of the model described in the article to the telecommunications industry. It must however be noted that sensitive customer information (such as names, age and gender) are not present at any time in the EPC network, meaning that working with subscriber CDRs (including bills or charges) solely will have to do here. This will most likely result in lowered accuracy of the model, but has the added benefit of not potentially infringing upon customer privacy to the same extent.

Employing Latent Dirichlet Allocation for fraud detection in telecommunications [XG07]

Description: A generative statistical model is employed here in order to build subscriber profiles, with any significant deviation in the behavior of an individual assumed to be strongly correlated with fraudulent activity.

Method: The authors begin by identifying a set of features that can be combined together in the form of a user profile. The main data that the authors focus on is the CDR, which contains all the necessary information that are recorded when a user makes a regular call. The authors identified three sets of features based on the users' CDRs. The first feature is the frequency by which the user makes calls during a given period of time, such as morning, night, afternoon and so on. The second feature is based on the destination of the call, thereby identifying the type of service the user is contacting, whether it is national, local, mobile, Internet etcetera. The third feature is based on the length of the call. Based on these features, the authors used the Latent Dirichlet Allocation (LDA) algorithm in order to build a ranking system for the users' profiles, thereby establishing a norm for each user. Based on the ranking of each user's profile, the authors were able to identify activities that do not fit the user's rank, and based on how far they are from the user's rank, they evaluated whether it is a fraudulent user or not.

Results: The authors have ran experiments on real life CDRs which were captured over a period of two months. Their results seem to be encouraging as in their best experiments, they were able to identify fraudulent users with both accuracy and efficiency.

Application in EPC: The EPC network keeps track of a lot of data on the users' activities, including the CDRs. Therefore the use of the method that is discussed in this paper is not only applicable but also recommended given their accuracy in detecting fraudulent calls.

Subscription fraud prevention in telecommunications using fuzzy rules and neural networks [EHP06]

Description: This paper presents a two-part method of supervised learning for detecting potential subscription fraud, comprising of a classification module and a prediction module.

Method: The data comprised $10,000$ Chilean subscribers registered to an undisclosed operator and included CDRs. The first module consisted of classification using fuzzy rules, using the tree branch like structure in order to group existing subscribers in to four labels. These included normal, insolvent and two types of fraudulent subscribers. A MLP was implemented as the prediction module, where the module was meant to identify these automatic labels.

Results: Results were promising, as the predictive module managed to identify 56.2% of all the true fraudsters, with a false positive rate of only 2.4%. Most of these false positive cases simply belonged to the insolvent category, and the method was therefore not entirely incorrect in its prediction, though labelling insolvent subscribers as fraudulent ones might be somewhat harsh.

Application in EPC: The method shows the feasibility of identifying a significant amount of fraudulent users, which would be another tool of cost optimization for many operators. However, all types of data provided to the authors does not flow through the EPC network. Though it should be noted that the main bulk of the authors' analysis still built on the subscribers' CDRs.

4.1.4 Faults

Within telecommunication networks there is potential for bugs or faults occurring. One example would be a node relaying multiple subscribers' data that crashes due to overload or such alike. Failures within a network can range from smaller instances, where individual nodes crash, to larger ones, where the entire network crashes as a result of multiple crucial node failures.

A Machine Learning Framework for Detection of Sleeping Cells in LTE Network [ZISAD14]

Description: The goal of this paper is the automatic detection of so called Sleeping Cells within LTE networks. These are cells that are not providing normal services to the users. The idea is to identify these cells for either maintenance or reassignment of users to other active cells.

Method: The authors created a three-step framework by which they are able to identify and locate sleeping cells within the network. The first step involves taking measurement of each cell's performance, this is done by gathering data in the form of the cell's KPIs as well as time and location information. The authors divided the data into a training set which contains normal fault-free simulated data, as well as validation and test sets that contain simulated faulty operations. The second step involves detection of faulty and sleeping cells, this is done by first projecting the training data into a low-dimensional space using the multi-dimensional scaling (MDS) method. Then two detection algorithms are used on the training data which are (k-Nearest Neighbor) and Local Outlier Factor. After detection of sleeping cells is completed, the third and final step is to localize the sleeping cell. This is done using the users' reported position information.

Results: For the evaluation, the test data is divided into 30% validation, and 70% test sets. Then the area under the curve (AUC) method is used as a measure of evaluation. The proposed method is able to create two groups, one group whose KPIs are normal, and another group whose KPIs are faulty or resulting from sleeping cell scenarios. The authors note that the k-NN algorithm seems to be outperforming the Local Outlier Factor (LOF), because the former is a global anomaly detector which is more suitable to this type of problems.

Application in EPC: KPIs are popular within the EPC network, they exist for each main and support node, as well as for user equipments and eNBs, thus making it quite possible to use this method for the identification of not only sleeper cells, but also for problematic main and support nodes in the network.

Machine Learning Based Session Drop Prediction in LTE Networks and its SON Aspects [DVB15]

Description: The goal is the early prediction of session drops within LTE networks well before the end of a session.

Method: The authors begin by noting that a session which is going to be dropped is usually in bad conditions well before the drop. And by identifying those conditions it is possible to predict which individual sessions will drop early on. The data that is used is a combination of live cell trace logs, time series, as well as a group of event monitors that are listed in the paper. The authors use two algorithms to confirm their predictions, AdaBoost and SVM, over both baselines, and Data Time Warping (DTW).

Results: The authors claim that their best predictions of session drops, were achieved using the DTW similarity kernel method followed by a baseline AdaBoost over statistical descriptors. According to their results, they were able to accurately predict session drops more than seven seconds before the occurrence of the drop. They also suggest that it is possible to use the existing data that they have used for predicting the drops, in order to identify the reason behind each individual session drop, and that the most common reason behind session drops is uplink problems.

Application in EPC: The paper was done by researchers at Ericsson specifically for the LTE network, thus as expected, there are a lot of similarities between the data that the authors used, and those that exist within EPC. So it is possible to use the techniques covered in this paper within EPC, particularly with SVMs over DTW.

4.1.5 User Location

For the purpose of load balancing the eNBs, studying subscribers' geographical movements in telecommunication networks is a topic of interest. Analysis of the user distribution can therefore be a powerful tool of optimization for these networks, allowing for a higher number of subscribers or avoiding node crashes. Another usage would be for general demographics purposes, something that might interest operators.

Annotating mobile phone location data with activity purposes using machine learning algorithms [LJWC13]

Description: This article investigates the behaviour of users by looking at their activities, mainly via phone calls made, with the goal of annotating or tagging locations. Several supervised classification models were tested to do the analysis.

Method: The authors' investigation consisted of four steps. First they defined a comprehensive set of temporal variables that characterized the calls appropriately. The second step was to apply filter and wrapper techniques for feature selection. The third step comprised the actual implementation of machine learning algorithms, individual ones such as SVM, Multinomial Logistic Regression, decision tree and random forests as well as a fusion of the results from them all. The final step included post-processing of sequential information via a Bayesian method in order to extrapolate accurate inference results. The data set the authors worked with contained more than one year worth of data from 80 subscribers.

Results: The ensembles of the models had a prediction accuracy of 69.7% accuracy, with the post-processing yielding an additional 7.6% improvement. The fusion model turned out to decrease predictive performance, highlighting that individual classifiers can be better suited for different specific problems occasionally. Noted by the authors was that a strength of the method is the low cost collection and fairly simple data requirements.

Application in EPC: For user profiling and demographics purposes it could be interesting to utilize a similar method as employed in this paper. The EPC network has access to CDRs as well as subscriber locations, making it feasible to implement this here as well.

Predicting a User's Next Cell With Supervised Learning Based on Channel States [CMV13]

Description: Using supervised learning, the authors looked at the channel state information (CSI) and user handover history of radio or cell towers, otherwise known as eNBs. The end goal suggested by the authors is allowing for more efficient resource allocation.

Method: A multi-class SVM was employed onto historical data consisting of the eNBs a user passed through and the CSI at each of these eNBs for the individual user. The model is kept fresh by regularly renewing the training set with real-time information and re-training.

Results: The model was tested against both a Manhattan grid scenario and a real-world downtown area of Frankfurt. The model increased its accuracy as simulation time was extended, and reached 100% with only 60% of the CSI history. Interestingly, only 100 subscribers was needed to construct an accurate prediction model.

Application in EPC: The results of this paper shows promising potential for location prediction within telecommunication networks. Although the CSI technically comes from outside the EPC network, the eNB-IDs are certainly stored somewhere in the MME node's register of subscribers.

Predicting the Location of Mobile Users: A Machine Learning Approach [AAH+09]

Description: In this article, the authors propose a spatial context model that classifies user trajectories via supervised learning methods.

Method: Based off the four last historical positions, the authors attempted to predict the next position with instance based learning (IBk), the J48 implementation of the C4.5 decision tree and several other methods. These were done with and without the ensembles of voting, bagging and AdaBoost. Finally, the machine learning methods' accuracy levels were compared to existing methods. These were the Lempel-Ziv algorithm (LeZi-update) and Mobile Motion Prediction (MMP) methods.

Results: Results showed that the voting model, consisting of IBk and J48, and was the strongest prediction model. For all degrees of randomness imposed on a user the model significantly outperformed the existing ones, in the worst case by 10% (for no randomness imposed at all). The authors also make a remark about enhancing the model with adding more semantics and contexts, such as temporal, application and social ones.

Application in EPC: The EPC network already does some location prediction via the E-SMLC node, primarily with MMPs. This paper indicates that this support node can be heavily augmented with machine learning algorithms for location prediction.

4.1.6 Performance

For maintaining a high standard of quality in communications, it is vital to monitor the performance of the network for the TSPs. This includes ensuring that eNBs transmit signals strongly and consistently, while nodes in the EPC relay on signals as they should.

Automatic Discovery of Sub-Optimal Radio Performance in LTE RAN Networks [BHO16]

Description: The authors propose a system that analyzes the various components of an LTE network and provide insights to the operators in the form of alarms in cases of deviation from the learned normal behaviours.

Method: The proposed system involves large scale data collection from various sources such as network performance, radio planning systems as well as historical data for different networks. This is followed by a learning mechanism, which involves the creation of signatures that are combinations of data that mark the various normal behaviours within the network. A number of combinations (signatures) are created in order cover the various parts of the network. Two types of analysis are done on the signatures. The first is anomaly analysis, where the signatures are analyzed in order to identify abnormal behaviours that do not fit the expected signatures. The second is factor analysis, where signatures are used in combinations in order to establish ideal configurations that result in the best performance possible, which the authors have labelled golden configurations. Those can be used by the operators later on as reset points for the network for best possible performance.

Results: The proposed framework offers two types of results. The first involves insights to the operators for parameters and configurations that deviate from the normal behaviour of the signatures. The second involves combinations of best performing configurations of parameters and signatures. The experiments that the authors ran showed that the insights that are provided by the framework are useful for operators in identifying underperforming nodes within the system. The recommended golden configurations provided are also useful in establishing a form of an ideal back up point for the best performance the network can have, given the set of parameters.

Application in EPC: The authors of the paper did their work in the LTE network, specifically at Ericsson. It is therefore likely that they have used components and data that exist within the EPC network. The combination of parameters (signatures) that were used by the authors may not be suitable for the EPC network, but the authors show that this framework is generic enough that signatures of any types, even with parameters within the EPC network, can be used.

Self-optimization of capacity and coverage in LTE networks using a fuzzy reinforcement learning approach [RKC10]

Description: The authors introduce a solution for enabling self-optimization of coverage and capacity in the LTE network. The solution mainly uses reinforcement learning to achieve this.

Method: The authors analyze performance monitoring data from eNBs, which includes data such as channel passlosses, antenna gains, bandwidth, user equipments' receiver noise and so on. When transmitting data to and from eNBs, the base stations can interfere with each other in the signalling. To minimize this one can adjust the angles of the antennas, downtilting them, in order to get more precise aiming of the signals. However, too aggressive downtilting of antennas can lead to coverage problems in the border areas of the base stations. The machine learning technique the authors implement is a fuzzy inference system (FIS) coupled with a reinforcement based method in the form of Q-learning. The FIS part generates a rule base on the base station data of which the reinforcement layer picks out the best rule from.

Results: When comparing against existing fuzzy logic-based reinforcement learning techniques, such as the Evolutionary Learning of Fuzzy rules (ELF), the solution implemented by the authors has up to a 20% improvement in performance. The self-healing solution is fully distributed with no signalling overhead between base stations and requires no prior information or human intervention.

Application in EPC: All data worked with, and the final application, was purposed for the eNBs and UEs only. There is therefore no direct translation of the application use for the EPC network. However, the idea of a self-healing network can still be applied in the EPC context. For example, it could have a mechanism implemented for re-allocating nodes that are overburdened with payloading.

4.1.7 Internet of Things

The subject of the Internet of Things (IoT) involves the integration of various physical and virtual objects into future networks. This includes any existing devices such as smart TVs, household electronics, automated systems, vehicles and others. This means that future networks will be receiving staggering amounts of data which must be reacted upon immediately. Decisions on which of this data to keep, or ignore, and what to forward to a centralized authority will be required. Manual analysis and reaction to such large amounts of data is almost impossible, not to mention extremely costly. Therefore the use of machine learning algorithms is not only applicable but necessary with the field of IoT.

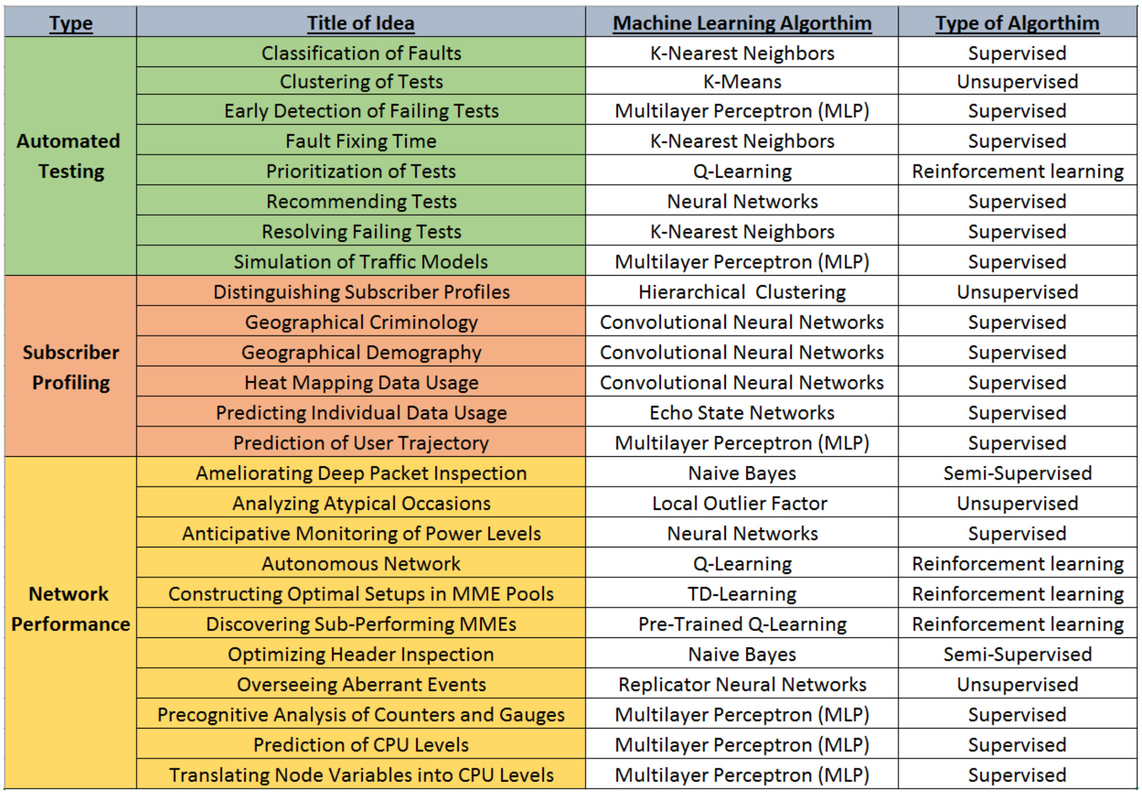
It is important to note that the term "Internet of Things" is a relatively new one. With its protocols still in development, research of machine learning within the field is still fairly scarce. However there are a number of examples that show how machine learning could be applied within the field of IoT, some of which are listed below.

- **Transportation**: Today's vehicles contain many sensors that keep track of both the car and the driver. There are already implementations of IoT, where all vehicles on the road are tracked and their sensor data are supplied in real-time to a centralized network. This includes tracking of vehicle's position, its speed, state of the driver and other useful data which the sensors pick up. Machine learning is then employed for data analysis, visualizations, prediction and optimization in order to provide continuous alerts to the drivers, updating road displays, dynamic allocation of buses, trams and subway timing, as well as providing assistance in case of emergencies. It is also expected that machine learning will play a vital role in the analysis of data from automated vehicles that are connected in self organized networks.
- **Healthcare**: Another example where IoT will be employed is within healthcare in order to keep track of patients through wearable and personal medical devices for services such as remote monitoring and consultation, management of chronic disease as well as programs for fitness and wellness. Machine learning is ideally used here for early detection of anomalies through analysis of historical records, as well as providing recommendations on new diets and fitness programs that are suitable for daily activities.
- **Home Utilities**: Various household utilities such as refrigerators, microwaves and ovens will be integrated into future networks for continuous monitoring. Machine learning is then used for historical usage analysis in order to save energy, demand and supply prediction for dynamical load-balancing and allocation, while providing constant updates and alerts to homeowners.
- **Manufacturing**: Cameras, controllers, sensors and gauges are a few of the data sources that exist within factories, which can be managed using an IoT system. The system could utilize machine learning algorithms for detection of anomalies, predictive maintenance as well as quality monitoring. This type of systems would be useful for automation of production, monitoring of equipments, remote diagnostics and early alerts of equipment failures.

4.2 Innovated Ideas

In the following subsections, ideas for application of machine learning within the EPC network are presented. The initial number of ideas that were innovated was 34. However, a number of those ideas were removed due to lack of data, contacts and resources. Thus, the final tally ended on 25 documented ideas, which are divided into three categories as shown in Figure 4.2

Several ideas share characteristics with each other, in some cases attacking the same problem from different perspectives. In other cases they can be complementary to the degree that they could be combined together for performing a more complex task or provide a higher level type of solution.



Type	Title of Idea	Machine Learning Algorthim	Type of Algorthim
Automated Testing	Classification of Faults	K-Nearest Neighbors	Supervised
	Clustering of Tests	K-Means	Unsupervised
	Early Detection of Failing Tests	Multilayer Perceptron (MLP)	Supervised
	Fault Fixing Time	K-Nearest Neighbors	Supervised
	Prioritization of Tests	Q-Learning	Reinforcement learning
	Recommending Tests	Neural Networks	Supervised
	Resolving Failing Tests	K-Nearest Neighbors	Supervised
	Simulation of Traffic Models	Multilayer Perceptron (MLP)	Supervised
Subscriber Profiling	Distinguishing Subscriber Profiles	Hierarchical Clustering	Unsupervised
	Geographical Criminology	Convolutional Neural Networks	Supervised
	Geographical Demography	Convolutional Neural Networks	Supervised
	Heat Mapping Data Usage	Convolutional Neural Networks	Supervised
	Predicting Individual Data Usage	Echo State Networks	Supervised
	Prediction of User Trajectory	Multilayer Perceptron (MLP)	Supervised
Network Performance	Ameliorating Deep Packet Inspection	Naive Bayes	Semi-Supervised
	Analyzing Atypical Occasions	Local Outlier Factor	Unsupervised
	Anticipative Monitoring of Power Levels	Neural Networks	Supervised
	Autonomous Network	Q-Learning	Reinforcement learning
	Constructing Optimal Setups in MME Pools	TD-Learning	Reinforcement learning
	Discovering Sub-Performing MMEs	Pre-Trained Q-Learning	Reinforcement learning
	Optimizing Header Inspection	Naive Bayes	Semi-Supervised
	Overseeing Aberrant Events	Replicator Neural Networks	Unsupervised
	Precognitive Analysis of Counters and Gauges	Multilayer Perceptron (MLP)	Supervised
	Prediction of CPU Levels	Multilayer Perceptron (MLP)	Supervised
	Translating Node Variables into CPU Levels	Multilayer Perceptron (MLP)	Supervised

Figure 4.2: Detailed overview of the innovated ideas, including categorization and machine learning algorithm recommended. Please refer to the idea description for further details.

4.2.1 Automated Testing

Whenever updating old or introducing new features, substantial testing needs to be performed in order to ensure the viability of the change. The changes introduced into a network, such as the EPC, are always looking to improve its functionality in some way, whether it is to add a feature not seen before or optimize the workflow of an existing process.

There are different kinds of testing, functional and system amongst others. Functional testing ensures that the update performs the new task as it is intended to. System tests looks at how various components interact. For example, the successful updating of an old feature in the MME node might affect the communication with the EPG node, thereby causing other features to crash in the EPG node instead.

When running tests, huge log-files are produced. The log-files are filled with data relevant to a specific test case and indicates the progress of the test run. The data logged is typically in the form of items that are described in Section 2.2.4. A test case is typically not run individually, but repeatedly instead. When running multiple test cases, they form a group called a test suite, where the included test cases can vary. If one seeks to test the stability of a function for example, a test suite consisting of the same test cases can be run. In this way, a measurement on how consistent the function is given by how often the test case fails and of what reason.

Finally, test suites can be grouped together into what is known as regressions. A type example of regressions would be system tests, where multiple functions and their communications need to be examined.

Classification of Faults

Description: The goal is to create a machine learning model that can classify failing test cases based on the severity of the problem and the priority of dealing with it.

Method: The classification of the failing test cases is done via comparative analysis over historical data. The process of classifying failing tests is as follows:

- **Suitable data:** The data includes other test cases that have failed in the past and share characteristics with the current failing test case. Examples of characteristics are counters and events that are invoked by the test case. The data can also include the exact same failing test case, in the scenario where the test case has failed and has been reported in the past, thus labels of test cases would be suitable parameters. It is also possible to take into account failing test cases that belong to the same category, such as test cases of the same test suite, test cases that test the same feature or same functionality. Two important parameters that must also be included into this algorithm as outputs are the priority and severity level of the test cases.
- **Selecting a suitable algorithm:** Given the available data and the existence of historical records, it is only natural to choose a supervised training algorithm as it will provide the most accurate model for the classification of the faults. An ideal supervised algorithm to use in this case would be K-Nearest Neighbors, as the goal is to find the closest example from the historical records, and the inputted data would be vectors in a multidimensional feature space, each with its own class label.
- **Training and validating:** Training of the model is done by storing the extracted features as vectors and their suitable class labels into the model. For any given failing test case, the model would then assign a priority level (low, medium or high) and a severity level (critical, major, medium or minor), which is taken straight from the test cases that are nearest to the current failing test case. Those levels are the standard priority and severity levels that are usually assigned for each failing test case that is reported to ClearQuest.

Expected Results: Implementing this idea will result in a method that can classify failing test cases based on the levels of severity and priority, as pre-established by ClearQuest.

Benefits: Reporting test cases into a tool such as ClearQuest requires some experience in identifying the levels of severity and priority of the failing test. This idea will be useful in assisting testers in expediting the reporting procedures by recommending suitable levels for each failing test.

Clustering of Tests

Description: The goal is to establish a model which is able to cluster test cases that share similar characteristics.

Method: There are a number of parameters that can be used for the purpose of finding correlation between tests. The most useful parameters exist within the resulting test logs, which include counters, gauges and events that have been changed or have occurred during the execution of a test. An unsupervised clustering algorithm such as K-Means is most suitable here. The goal is to cluster test cases according to their characteristics, the characteristics being the aforementioned features. The training of the model involves inputting the extracted features into an optimal clustering representation with the use of the K-Means algorithm. Thus, tests that have similar characteristics or identical ones will be clustered together. The trained model is applied on the logs of all existing test cases and clusters will form as output.

Expected Results: Implementing this idea will result in a method that is able to create groups of tests that share similar or identical characteristics.

Benefits: System test involves large amount of test cases, a lot of which are similar, if not identical to each other. Running a test suite, or group of them, can easily mean running similar test cases repeatedly, which is a waste of time and resources. This method allows for finding those tests that are similar and clustering them into groups, thereby running one test out of the group would be the same as running all of them.

Early Detection of Failing Tests

Description: The goal is to establish an automated process which is used for the early detection of failing test cases and test suites, during extended live tests and regressions, as shown in Figure 4.3.

Method: The process of detecting the failing tests during regression is as follows:

- **Feature extraction:** There are many parameters within the logs of a test and it would be impossible for a machine learning algorithm to extrapolate patterns from all of them. It is thus necessary to extract certain parameters from the logs which are useful for the training of the algorithm. Based on recommendations from the designers of those tests, three types of parameters should be extracted from the logs. Those parameters are counters, events and signals. It is important to note that these parameters are ideal for the creation of a proof of concept of the idea. However, full implementation of the idea will likely require the need of more parameters which can be extracted from those log files.
- **Selecting a suitable algorithm:** The most suitable type of machine learning algorithms for dealing with this type of problems is a supervised method. Tests within the EPC network are run repeatedly every time

there is a new change in a node's behaviour, and all logs tend to be saved for future reference. There are thus plenty of logs that can be used for cross referencing when training a machine learning algorithm, as historical data. There are a number of supervised methods that can be used here, most suitably MLPs. This is due to the fact that the model will be be receiving multiple inputs and producing one output (whether the test case will pass or fail). However, other supervised approaches can also work, such as a Naive Bayes Classifier.

- **Training and validating:** The training of the model will take place outside of the regression or live testing. This is because, the training phase requires some processing power that could potentially affect the running of the test cases. As mentioned before there are plenty of logs for each test case that can be used as historical data, these logs are split into training and validation sets. The training of the algorithm will make use of the features that were extracted from the logs, as inputs to the model. The model will then produce an output that indicates whether the test case will pass or fail. In terms of validating the model's results, the output is matched against validation set's results and adjustments to the model are done in cases of mismatching results, thus the model is improves the more logs that it is trained with.
- **Running the model:** During regressions or live testing, the currently running test cases will produce their own logs. The model is then run concurrently in a process that takes care of accessing and extracting the features from the logs, inserting them into the model. The model would then evaluate the extracted features and make the decision on whether the test case will pass or fail. If the model predicts that the test case will fail, it should raise an alarm indicating as such. Otherwise the model continues to run concurrently along with the running of the test case.

Expected Results: Implementing this idea will result in a classifier that can raise alarms in the cases of predicted failure of test cases, during live regressions.

Benefits: System testing and is a long and expensive processes that is done every time a new feature is added to one of the nodes in the network. Modifications of the regular tasks of the node require a new full scale test in order to verify that those changes will not affect the system in a negative way. Instead of waiting until the end of the test in order to find out whether the test has passed or failed, this process will alarm the testers about possible failures early on. A process that can predict the failure of those tests at an early stage is valuable for saving both time and resources that are spent on these tests.

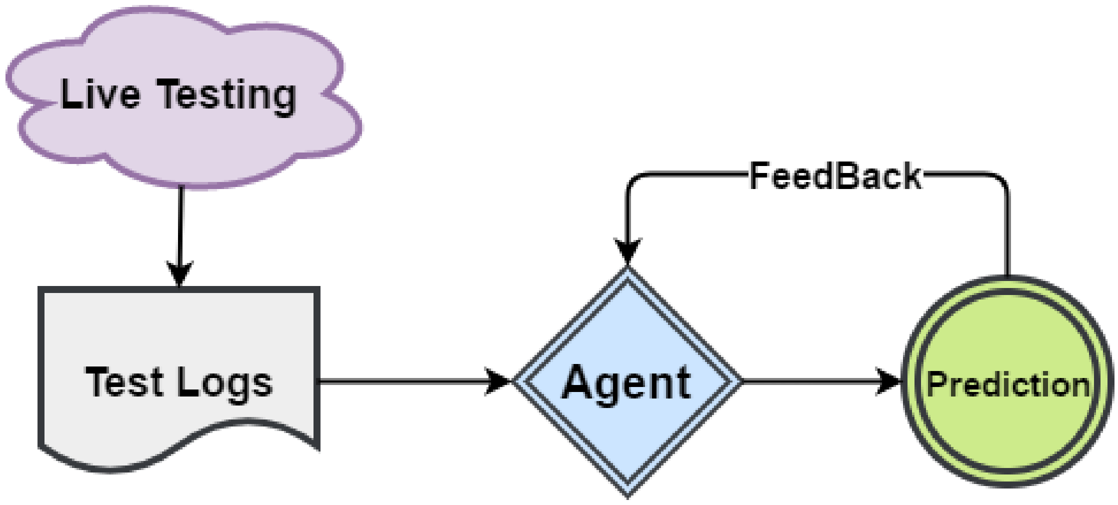


Figure 4.3: Overview of the workflow of an agent working in tandem with the live testing.

Fault Fixing Time

Description: The goal is design a method by which it is possible to estimate the amount of time needed to fix faults.

Method: This idea involves comparative analysis of the existing historical data. It works as follows:

- **Suitable data:** Faults have a tendency of repeating themselves in different features and functions within a system, thus it is useful to take into account existing faults that share characteristics with the current failing test case, such as similar counters, events and signals. For this idea to work, it is important to take into account a parameter that represents the time needed to fix past faults. Many tools that are used for reporting faults, such as ClearQuest, include such a parameter.
- **Selecting a suitable algorithm:** Supervised algorithms such as K-Nearest Neighbors work well in this case, as there are plenty of existing records to use for training and cross referencing.
- **Training and validating:** Training makes use of the extracted features from the existing faults, including the parameter of time. It is worth noting that even in the cases of similar faults, the time needed for fixing them might differ slightly as other factors and circumstances come into play. Thus, for the evaluation of the fault it is best to estimate the time needed to fix the fault by averaging over the time of the closest neighboring faults. The estimations will become more accurate the more faults are inputted into the algorithm, and will eventually reach optimal time for fixing a new fault.

Expected Results: The solution will result in a feature that allows for the estimation of time costs needed to fix faults within the nodes of the EPC network.

Benefits: Faults are reported on daily basis from the various nodes of the network, resulting in a huge list of faults that require the attention of developers. This idea allows the developers to optimize their time, as they would be able to choose which faults to fix based on both their priority and the time needed to fix them.

Prioritization of Tests

Description: The goal is to establish a method by which it is possible to prioritize test cases or test suites during live testings or long term regressions.

Method: The process of allocating and prioritizing test:

- **Feature extraction:** The testing portal runs the tests in the form of a First Come First Serve, unless the admin manually prioritized particular tests, in which case those tests are marked as such. The portal contains plenty of logs and information for each test, such as the test's period, results, activities, and resources in use. All those information can be used as parameters for the algorithm.
- **Selecting a suitable algorithm:** Reinforcement learning is likely to be the most suitable choice of algorithms to be used here. This is because, the features that are used to train the algorithm are circumstantial and can change depending on the manual inputs that the admin makes.
- **Training:** The training of the model will take place outside of the portal with the extracted features as training parameters. This is because training of the model in reinforcement learning is a heavy process that might affect other running processes.
- **Running the model:** The model is an agent that is placed as a process within the portal, which takes care of prioritizing tests based on the time needed and the time available for tests, as well as the amount of resources that the test needs. While the model attempts to make decisions of prioritization based mainly on the aforementioned parameters, the admin will also play the role of adjusting the agent's choices by indicating whether those test cases are OK to be prioritized or not.

Expected Results: Implementing this solution will result in a method that evaluates and prioritize tests in an efficient manner.

Benefits: According to testers within the EPC network, the current test portals are not efficient in allocating and prioritizing tests. By using machine learning, this model is able to identify and prioritize tests in an optimized and efficient manner which allows for saving both time and resources of the portal.

Recommending Tests

Description: The goal is to create a tool that recommends the necessary tests that need to be applied when alterations occur on the code.

Method: The idea is to establish a form of correlation between newly introduced features/updates, and the tests that are applied to them. The process of building such a model is as follow:

- **Feature extraction:** The most suitable data that can be used for building a correlation model includes information on the type of alteration, such as if it is a new feature, small update or fixing of bugs amongst others. There also need to be some form of record kept with regards to which test cases are applied when these alterations take place, or, better yet, which test cases fail when the alterations are introduced.
- **Training the model:** The most suitable type of machine learning here is supervised. The correlation model would take the aforementioned, extracted features. Since updates and tests are done repeatedly, this would be a functional method for cross referencing.
- **Running the model:** Whenever a new update is introduced, information on the update is inserted into the model, which would then provide a recommendation of suitable tests that need to be applied on the update. It is worth noting that the model is not self-correcting. Thus in cases where the model provides wrong recommendations, the model must be adjusted manually by removing those recommendations or adding new ones.

Expected Results: This solution results in a model that is able to provide recommendations on which tests need to be applied when an update or a new feature is introduced.

Benefits: Function testing is done every time a change occurs in the code, whether if it is for a new update, a new feature or a simple bug fix. There is a huge number of test cases that are applied with each new change, thus the benefits of having a tool that can reduce the number of tests that need to be applied are quite valuable in terms of saving time and other resources.

Resolving Failing Tests

Description: The goal is to have a list of recommended solutions or steps that could be followed in order to resolve a failing test case.

Method: Failing tests indicate problems with the new features and updates that are introduced. The process of fixing those faults are documented thoroughly in multiple ways. For example reports of the faults on ClearQuest are updated whenever the status of the fault is changed, and when the fault is fixed the process by which the fault is fixed is also noted in the report for future reference. Another example includes the updates on the Git repository, when a fault is fixed and committed to Git, a suitable comment is included that describes the fix. Using the above knowledge along with other information, such as the name of the failing test case, which test suite it belongs to and the reason why the test failed, it is possible to establish a a model that uses historical records for recommending solutions to faults that share similar characteristics with past faults.

Expected Results: A tool that can be used for searching for recommendations on fixing faults within the network.

Benefits: Many of the faults that are reported have a tendency of repeating themselves. New engineers that attempt to fix these faults may find it handy to have a tool that can provide recommendations on how to fix those faults. The tool can be taken one step further and be used to apply the solution directly under the supervision of the engineer.

Simulation of Traffic Models

Description: A model that simulates behaviors of various nodes in a network. Particularly useful on a system test level, when one seeks to test the functionality of one node, given data traffic sent to it from other nodes in the network.

Method: Using historical data logged in the networks used by telecommunications operators, a supervised learning approach can be taught to emulate the behavior of nodes, as in the quantities and types of data being outputted. From this, several behavioral profiles of network traffic can be constructed for each node. MLPs would work best for approximating the non-linear outputs from a node.

Expected Results: A model for approximating node behavior in different settings. These settings refer to various types of behavioral profiles, such as low-key weekday nights, intense seasonal celebrations or regular working days traffic.

Benefits: Instead of handcrafting traffic models or creating complex virtual structures that replicate the hardware behavior of a node, the machine learning model is intended to be a efficient method that saves computational resources during simulations in the long run.

4.2.2 Subscriber Profiling

It is of great interest to telecommunications operators to track individual user trends. Using machine learning to spot trends in subscriber behaviors and demographics can help optimize the operators' cost efficiency, both by reducing redundant services and preemptively know which nodes in the network need prioritized attention.

Distinguishing Subscriber Profiles

Description: A method for detecting profiles of user types. Examples of profiles could be a modern, office worker living in the cities versus an industrial laborer that lives on the countryside.

Method: Data needed from the EPC network includes a whole set of counters, gauges and events taken from both the MME node and the CDRs found in the EPG part of the EPC network. Unsupervised algorithms are advised, K-Means and Hierarchical Agglomerative Clustering. Trying out both algorithms would be recommended since they approach the clustering task from different directions, top-down and bottom-up, respectively. In doing this, one could avoid the pitfall of mislabelling outliers in the data set by having enough clusters to account for outlier classes as well.

Expected Results: Labels that represent classes, assigned to each subscriber. A potential issue could be, as mentioned before, assigning a subscriber to a category they do not belong to.

Benefits: The clusters could potentially yield some previously unknown information regarding behavioral patterns with users, showcasing the different types of subscribers utilizing LTE services. This information can be used for other machine learning applications for more powerful prediction analyzes or packaged as a product that both TSPs can sell to the operators.

Geographical Criminology

Description: Creating discretized heat maps over regions by the amount of blacklisted devices registered.

Method: Unique UEs, based on their IMEIs, are tracked and combined with the reported locations of the UEs, all of which can be extracted from the CDRs in the EPG part of the EPC network and the E-SMLC support node. In addition, the equipment status is checked with the EIR support node. Since the full map is a grid, the UE locations need to be translated into a grid position. For the machine learning phase, supervised approaches are recommended. Inputs are defined as the UEs' locations in the grid, the status of the equipment used and the recorded time, outputs as the number of users. Suitable algorithms are ANNs, such as MLPs, with an outlook for a more complex mapping done with Convolutional Neural Networks.

Expected Results: The model should work as a database of heat maps, where each time step contains a heat map over blacklisted UEs in the regions mapped. See Figure 4.4 for an example of such a heat map.

Benefits: As a statistical tool, the model can assist law enforcement in tracking and analyzing criminality in regions mapped. It should be noted that the criminality here pertains only to the stealing of mobile devices and not any other types of crimes. This idea has a implementation similar to the idea of mapping user distribution in different regions.

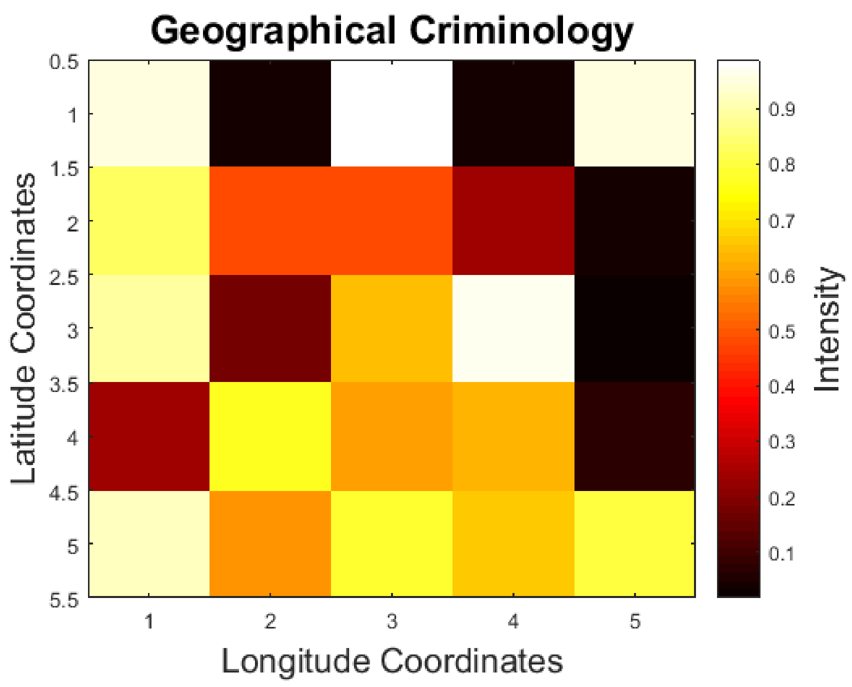


Figure 4.4: Example of a discretized heat map, where the intensities refer to ratio of blacklisted UEs.

Geographical Demography

Description: Creating discretized heat maps over regions by the amount of users present in the areas.

Method: Unique UEs, based on their IMEIs, are tracked and combined with the reported locations of the UEs. This data can be extracted from the CDRs in the EPG part of the EPC network and the E-SMLC support node. The locations need to be translated into a grid position, since the full map is a grid and thereby discrete. For the machine learning phase, supervised approaches are recommended. Inputs are defined as the UEs' locations in the grid and the recorded time, outputs as the number of users. Suitable algorithms are ANNs, such as MLPs, with an outlook for a more complex mapping done with Convolutional Neural Networks.

Expected Results: The model should after training and validation work as a database of heat maps, where each time step contains a heat map over subscribers in the regions mapped. See Figure 4.5 for an example of such a heat map.

Benefits: TSPs can use the model as a statistical tool for analysis in order to improve the distribution of eNBs in the mapped regions. This idea is analogous to the idea of mapping data usage distribution in different regions. Another benefit of the model is for analysis of population dynamics in a geographical area.

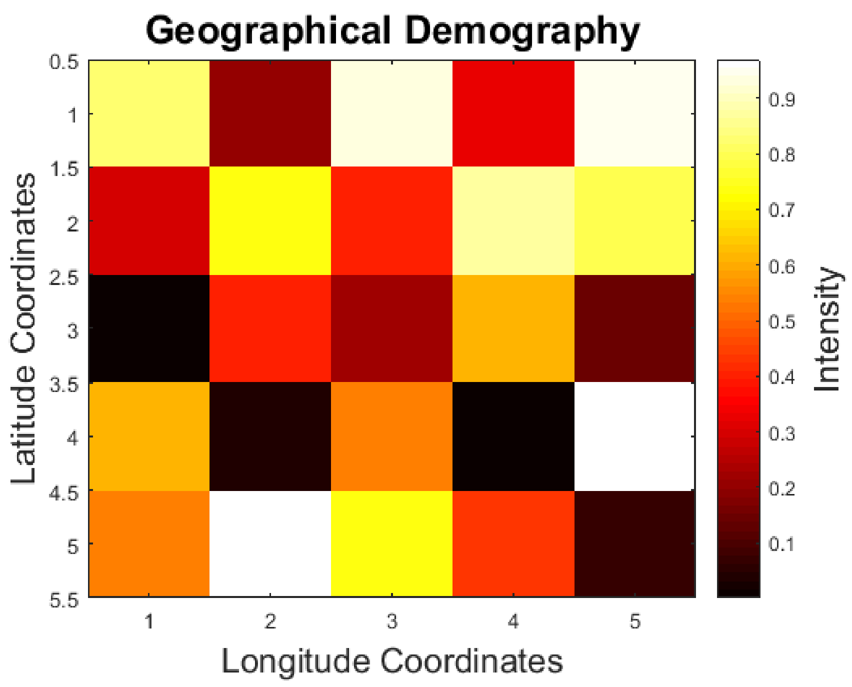


Figure 4.5: Example of a discretized heat map, where the intensities refer to ratio of UEs.

Heat Mapping Data Usage

Description: Creating continuous heat maps over regions by the amount of data flowing through the areas. Here, data refers to what is payloaded through the EPG nodes to the subscribers.

Method: The OCS support node handles throughput of data to UEs, this data throughput is combined with the reported locations of the UEs. All this can be extracted from the CDRs in the EPG part of the EPC network and the E-SMLC support node. For the machine learning part, a supervised approach is recommended. Inputs should be defined as the UEs' locations and the recorded time, outputs as the data usage. Suitable algorithms are ANNs, such as MLPs, with an outlook for a more complex mapping done with Convolutional Neural Networks. However, one must account for the lack of data used in various regions. This means that the training set of the model needs to include locations where no data is being used. Potentially, these data points need to be handcrafted by a domain expert. The number of these "no data usage locations" should be in the same order as the number of users, at least. A recommended starting point would be to equal it with the number of users.

Expected Results: The model should after training and validation work as a database of heat maps, where each time step, as defined by the user, contains a heat map over the data usage in the regions mapped. See Figure 4.6 for an example of such a heat map.

Benefits: TSPs can use the model as a statistical tool for analysis in order to improve the distribution of eNBs in the mapped regions. This idea is analogous to the idea of mapping the user distribution in different regions.

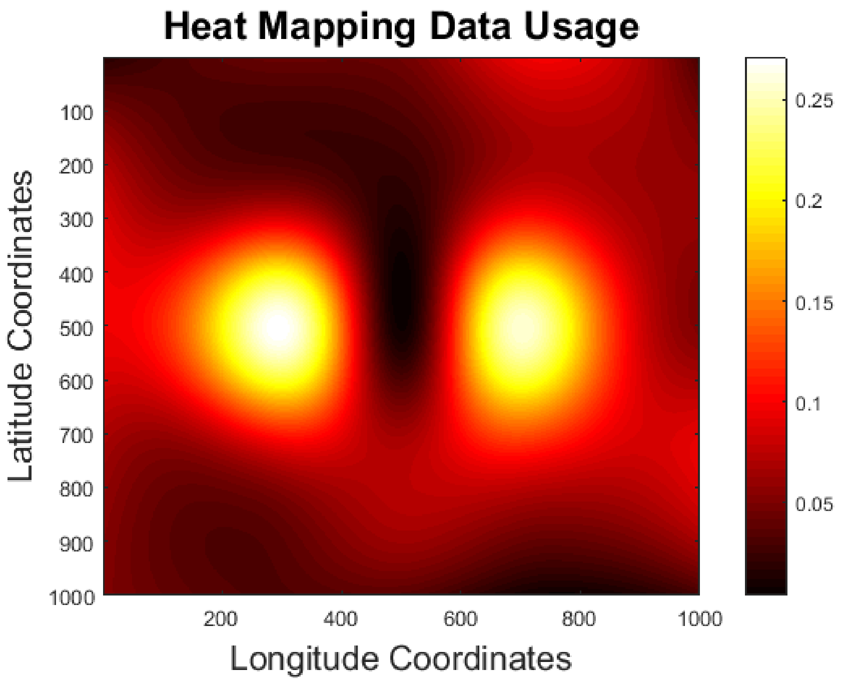


Figure 4.6: Example of a continuous heat map, where the intensities refer to data usages in any location.

Predicting Individual Data Usage

Description: A model for predicting the amount of data a subscriber will use.

Method: Using historical logs of how much data a subscriber used previously. All this information can be extracted, in the form of counters and events, from the CDRs generated in the EPG node. Taken as inputs should be data packet amounts and sizes for one subscriber over the past t_{before} time steps, a parameter that needs to be tuned in during validation and re-training phases. The model is therefore also individual to each subscriber, though doing this for groups of subscribers should be possible as well. In such case, subscribers would have to be pre-labelled, either via an unsupervised machine learning method, which is described in a different idea, or be defined by information given by telecommunications operators. Outputs should be defined as the data usage by the relevant subscriber in each of the next t_{after} time steps. t_{after} should be selected depending on how far ahead in time one wishes for the model to predict. Supervised approaches to dealing with the prediction of data usage are recommended, such as MLPs and Echo State Networks with continuous outputs.

Expected Results: The model can predict the data usage in the short-term future for individual users, using historical data.

Benefits: Predicting the data usage profile of a user means that said user can be assigned to specific EPG nodes. For example, a subscriber often utilizing streaming services can be offloaded to an EPG node dedicated to handling large bandwidths of data payloading. Knowing the data usage beforehand can also serve the purpose of load balancing both eNBs as well as EPGs.

Prediction of User Trajectory

Description: Detect user trends in daily movements using their historical positions. Positions are expressed in geographical coordinates, these are tracked for all UEs via the E-SMLC support node during c-plane signalling.

Method: There are two ways of achieving this, a discretized and a continuous version. The first involves forming a 2D grid of the physical map, and classifying movement directions in each time step. The second does the predictions with exact predictions, including continuously defined directions and step lengths. A supervised approach fits best, an Echo State Network or MLPs are recommended here. Depending on which of the methods one goes with, discrete or continuous outputs, respectively, are recommended. For the inputs, the positions over the last t_{before} number of time steps must be considered when utilizing MLPs. The tuning of this parameter will be subject to the optimality of the implementation. Outputs should be defined as the predicted positions for each of the next t_{after} time steps, a different parameter that should be chosen depending on how far in to the future one wishes to predict the user trajectory. Historical positions can be fetched from the E-SMLC support node and the model should ideally be constructed per UE, but could also be feasible for groups of subscribers in a more generalized form. In the case of having subscriber groups, subscribers would have to be pre-labelled with affiliations and coordinates should be normalized so that subscribers move in their own reference frame. Pre-labelling can happen via an unsupervised machine learning method, which is described in a different idea, or be defined by information given by telecommunications operators. The constructed model can then be validated as described in Section 2.1.1, with a specific validation set.

Expected Results: The model can predict the trajectory for individual users, as seen in Figure 4.7. The E-SMLC node already does a simple and inaccurate calculation of user trajectories in the EPC network, but the machine learning model is meant to be more accurate.

Benefits: The model can initially augment position prediction utilized in the E-SMLC node today, but long-term replace it altogether. This improved accuracy in location prediction means that the LTE network can react better when preemptively attempting to load balance eNBs in the geographical region.

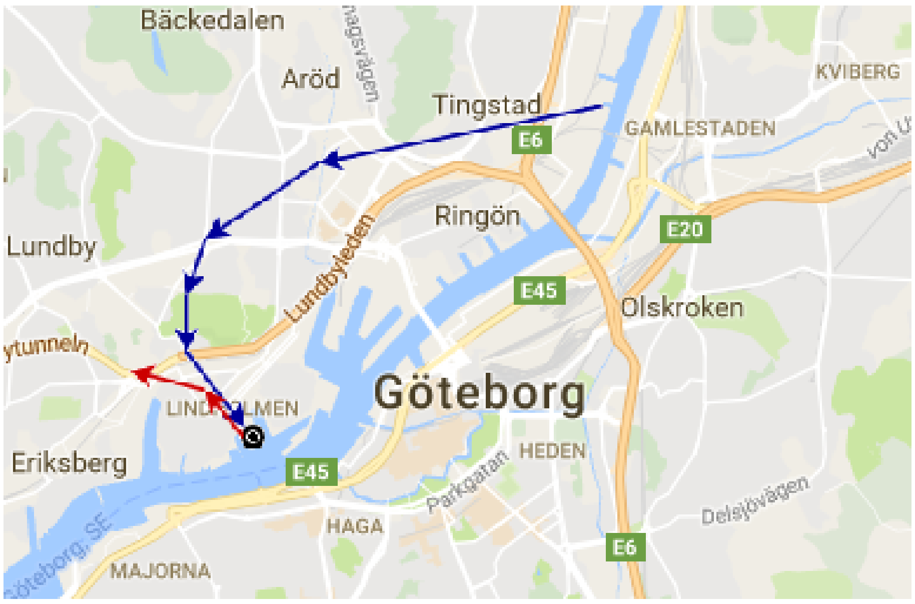


Figure 4.7: Example of a machine learning algorithm predicting a user's trajectory. User is denoted by black circle, previous movements by blue arrows and predicted movements by red arrows. Here, $t_{before} = 5$ and $t_{after} = 2$, background courtesy of Google Maps.

4.2.3 Network Performance

Monitoring the performance of the EPC network is of high priority to operators in the telecommunications industry since latency is costly for them. Subscribers are paying for uninterrupted access and therefore expect to be able to make calls as well as having Internet access at any given time of the day, resulting in downtime being a source of frustration for users whenever it occurs.

Ameliorating Deep Packet Inspection

Description: A tool that identifies type of service packets belong to by looking at the packet bodies.

Method: Data packets transmitted in the EPC network contains substantial information that can used as inputs for a supervised algorithm, which then maps said inputs to outputs in the form of what service the packet belongs to. Recommended algorithms are MLPs and Naive Bayes for classification. It might not be feasible to use the full packet as input, since this most likely correlates to longer computing times for the machine learning algorithm. A smart feature selection performed by a domain expert may be needed for the identification of useful input parameters to extrapolate from the packet bodies.

Expected Results: A model that can sort packets into services by looking at the full body of packets. There are already tools today being deployed to perform Deep Packet Inspection (DPI). They typically follow a set of simple, logical rules that take a long time to check. Ideally, the machine learning model described here works faster than existing deterministic methods, while maintaining or improving the accuracy.

Benefits: The identification of which services packets belong to enables the creation of traffic flows. These service based flows can be allocated through nodes specific for different types of payloading. This is most relevant to the EPG domain, where there are typically several working together in a pool. Some EPGs can then be dedicated to streaming services over Skype for low latency, while other EPGs are instead dedicated to handling lower bandwidth demanding browser surfing, for example. Another benefit is identifying packets that is spam data and thus preventing them from constraining subscribers or the network.

Analyzing Atypical Occasions

Description: A tool for predicting special events in the world, based on divergent CPU behavior in a node.

Method: Data in the form of historical CPU levels of a node are used for the training of an anomaly detection method. To this end, the unsupervised methods of Local Outlier Factor and Replicator Neural Networks are recommended. Local Outlier Factor looks at the densities of the sample set, a form of a clustering analysis. Replicator Neural Networks attempts to compress and then decompress an input in order to see what the reconstruction error is. In this case, one only wants to feed the algorithm “normal” data during training, in a sense the method is more semi-supervised. The model takes training inputs in the form of historical CPU levels, in a typical looking time period. The final model is validated against new data, more specifically data known to be outlying in order to check whether the behavior converges to or diverges from normal behavior.

Expected Results: A model that can tell whether a set of future CPU levels in a node correlate to an accurate prediction by a predictor. This is done by analyzing the most recently recorded data in order to see whether it converges or diverges from normal behavior. This analysis can therefore say whether the inputted data to such a predictor is credible enough to do predictions with.

Benefits: This model can be used to complement existing tools that predict future CPU levels, specifically in the case when the input data is atypical. Some special events are regular, such as seasonal celebrations, others are irregular. These include large scale accidents or disasters that happen rarely.

Anticipative Monitoring of Power Levels

Description: The goal is to design a model that is able to detect peaks of power levels that may be dangerous to the node.

Method: Each node contains a number of parameters that keep track of various activities of the node. These parameters are stored in log files that are regularly updates by the node. Certain parameters that are useful for this idea are counters and gauges that maintain the current power levels of a node. Using a neural network that takes in the labelled values of the counters and gauges, it is possible to create a model that can fit a suitable function for the data. The function's curve is then used to predict whether or not the power levels will reach dangerous peaks. The neural network is trained with features that are extracted from offline logs that have been generated in the past, then applied on the logs that are generated through the live execution of a node.

Expected Results: A model that is able to establish normal power levels of a node, detect any abnormal behaviours and possibly dangerous peaks.

Benefits: Early detection of peaks in the network is useful for taking necessary steps to prevent damages to hardware and therefore any negative effects on the network.

Autonomous Network

Description: The goal is to establish a method that monitors the network and allows for an agent to take preventive actions when problems arise.

Method:

- **Feature extraction:** There is plenty of data that could be used for keeping track of user activities within the network. The most important ones include, user traffic and data usage per user, both of which are given as events in the network. Other parameters that could prove useful are counters and gauges that keep track of the various changes within the network. The information is logged in real-time and stored in databases for future reference.
- **Selecting a suitable algorithm:** Reinforcement learning is a suitable approach in here, as it will be able to make use of both the historical records for training the agent, and the real-time feed of data for further adjustments of the agent.
- **Training the model:** Training of a reinforcement learning algorithm is a costly and lengthy process, hence it can be sped up with pre-training before deploying the agent. This is where the historical data would be useful as it could be used as inputs for training the model.
- **Running the model:** The trained model is run as a process within the network which reads the new data and produces decisions on whether the network will be overwhelmed with users or not. The agent's actions are adjusted through a reward system that indicates whether the results are suitable or not. The output of the model is treated with a percentage in order to indicate the accuracy of the prediction, the more training the agent receives the more accurate it becomes.
- **Reactions to the model:** The model's output will serve as a warning for cases where there will be an overload of users on the node. Based on the output, the node's admins can make decisions on whether they should

be moving users to different nodes or assign more resources to the node. In the cases where the model's output proves to be false, the model should be adjusted with a negative reward which updates its learning curve for more accurate future decisions.

Expected Results: The implementation of this idea will result in a method that can be used for the detection of peaks of users in the network.

Benefits: Early detection of peaks in the network is useful for taking the necessary steps to prevent any negative effects on the network. It is possible to, for example, scale the resources available in the node by adding or removing them, alternatively throttling the threshold of users to prevent overloading the node.

Constructing Optimal Setups in MME Pools

MME Pool: In the EPC network, a group of MMEs that work to service users in different areas are called an MME Pool. The goal is to share the workload between the MMEs and to prevent denial of service when an MME becomes unavailable.

Description: The goal of this idea is to create a model that is able to predict the most optimal setup of users that will be attached to available MMEs in an MME Pool area.

Method:

- **Feature extraction:** Naturally, the most important parameters to take into account for this idea include number of available MMEs in the area, the number of connected users in each MME, and the number of new users that are connecting at each period of time.
- **Selecting a suitable algorithm:** Reinforcement learning would be ideal for this idea, as it will be able to make use of both the historical records for training the agent, and the real-time feed of data for further adjustments of the agent. Training of the agent takes place in offline mode by making use of historical data such as the ones that are discussed in feature extraction. The model will be able to establish patterns where an MME has a suitable number of users attached to it.
- **Running the model:** The trained model is then run as a process within the network which reads the constant change of newly connecting and disconnecting users within certain period of time, taking into account available MMEs in the pool and the amount of users connected to them. The model then outputs a suitable number of users that should be connected for each MME. The agent's decisions are updated within each change in the environment "network", such as MMEs becoming unavailable, user reassignments and increasing number of connecting users. The agent's actions are adjusted through a reward system that indicates whether the results are suitable or not, this can be done by supervisors that keep track of the progress of the agent.

Expected Results: A model that is able to device an optimal setup of the number of connecting users to the network, against the number of available MMEs in the pool.

Benefits: Currently, the MME pool works by an algorithm that only takes into account connecting users, and attempt to have the MMEs in the pool share the workload. This idea provides the ability of optimal use of each MME in the pool, reducing overload on MMEs and potentially increasing efficiency of MME reception of new connecting users, even during peaking periods of the day.

Discovering Sub-Performing MMEs

Description: The idea involves early detection of failing or sub-performing MMEs in an MME pool in order to take suitable preventative actions.

Method: An MME produces plenty of logs that reports on the status and health of the node. Health checks are usually monitored using counters and gauges, but other parameters within those logs can also play a role in making the model's results more accurate. A pre-trained reinforcement model is suitable for this idea because it can make use of past data that was produced by the MMEs for training and establishing patterns of normality for the state of an MME. The trained agent would then be put as a separate process within the network and make use of the live logs that are produced by each MME in order to detect abnormal behaviours within the node and detect whether or not the node is sub-performing. Several processes can be spawned for multiple independent agents that monitors each MME node separately.

Expected Results: A model that is able to detect abnormalities within an MME node and detect whether the node is sub-performing or failing early on.

Benefits: An important task within MME is stability and availability, users that are connected to a node that could potentially fail would lose service resulting in huge potential costs to the company. Early detection of failing MMEs would give the network the ability of re-assigning the connected users to other available MMEs within the node thereby avoiding the anticipated problems.

Optimizing Header Inspection

Description: A tool that identifies type of service packets belong to by looking at the packet headers.

Method: Data packets being transmitted in the EPC network have headers, which is essentially superficial information of the packet. This includes IP-addresses and port addresses amongst others. These header parameters can be used as inputs for a supervised algorithm, which then maps said inputs to outputs in the form of what service the packet belongs to. Recommended algorithms are MLPs and Naive Bayes for classification.

Expected Results: A model that can sort packets into services by looking at superficial packet information, also known as packet headers, see Figure 4.8.

There are already tools today being deployed to perform header inspection. They typically follow some set of simple and deterministic rules, which works quickly, but not necessarily accurately. Ideally, the machine learning model described here can replace existing methodology and augment the header inspection with a higher accuracy rate.

Benefits: By identifying services packets belong to, traffic flows can be created. These service based flows can then be allocated through nodes specific for that type of payloading. This is most relevant to the EPG domain, where there are typically multiple ones working in a pool. Some EPGs can then be dedicated to streaming services for low latency, while other EPGs are instead dedicated to handling lower bandwidth demanding browser surfing, for example. Another benefit is identifying packets that is spam data and thus preventing them from constraining subscribers or the network.

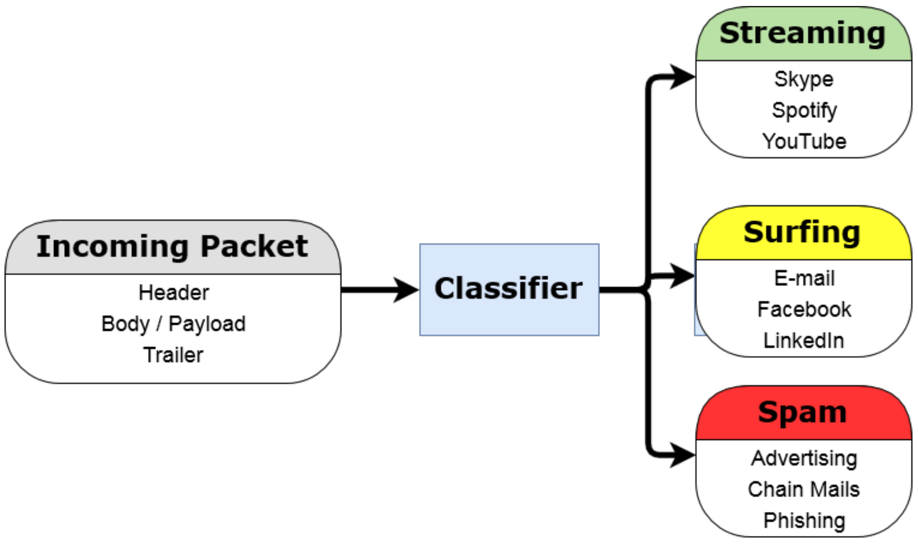


Figure 4.8: The machine-learned classifier in action, analyzing packet headers and sorting the packet in to the respective category of service.

Overseeing Aberrant Events

Description: A tool for detecting unusual counter and gauge values.

Method: The counters and gauges are used for the training of an anomaly detection method. Unsupervised methods such as Local Outlier Factor and Replicator Neural Networks are therefore recommended. Local Outlier Factor looks at the densities of the sample set, a form of a clustering analysis. Replicator Neural Networks attempts to compress and then decompress an input in order to see what the reconstruction error is. In this case, one only wants to feed the algorithm "normal" data during training, in a sense the method is more semi-supervised. The model looks at the changes in counters and gauges of a node for training, using only values from a typical looking time period. The final model is then validated against new data, more specifically data known to be outlying in order to check whether the behavior converges to or diverges from normal behavior.

Expected Results: A model that can tell whether a set of future node variables correlate to an accurate prediction by a predictor. This is done by analyzing the most recently recorded data in order to see whether it converges or diverges from normal behavior. This analysis can therefore say whether the inputted data to such a predictor is credible enough to do predictions with.

Benefits: This model can be used to complement existing tools that predict future node variable values, specifically in the case when the input data is atypical. Some special events are regular, such as seasonal celebrations, others are irregular. These include large scale accidents or disasters that happen rarely.

Precognitive Analysis of Counters and Gauges

Description: A tool for predicting the changes of counter and gauge values, or node variable values, based on their history.

Method: Utilizing historical values of the counters and gauges in a node, a supervised learning approach can then learn to predict future values. MLPs are recommended, since this essentially is a function fitting problem and non-linear relations are likely to be mapped. The idea should be implementable in any of the main and support nodes in the EPC network. Taking the recorded node variable values from the t_{before} last number of time steps as inputs, the machine learning model can output values for the t_{after} next time steps.

Expected Results: This tool could ideally be used for analysis and monitoring of the node in which the model is deployed in.

Benefits: Predicting the behavior of a node is essential for optimizing its performance. Thus, this model can provide an additional tool for analyzing the node's role in the EPC network.

Prediction of CPU Levels

Description: A tool for predicting the CPU levels in a node, based on its historical values.

Method: Utilizing historical values of the CPU level in a node, a supervised learning approach can then predict future values of the CPU level. MLPs are recommended, since this essentially is a function fitting problem and non-linear relations are to be mapped. The idea should be implementable in any of the main and support nodes in the EPC network. Taking the recorded CPU levels from the t_{before} last number of time steps as inputs, the machine learning model can output CPU levels for the t_{after} next time steps.

Expected Results: This tool could ideally be used for analysis and tracking of trends in the CPU load of a node.

Benefits: Knowing beforehand how much a node will be burdened is useful for power saving purposes. This knowledge can also be used for a worst case scenario maintenance and repairing of a node that is unique in its network.

Translating Node Variables into CPU Levels

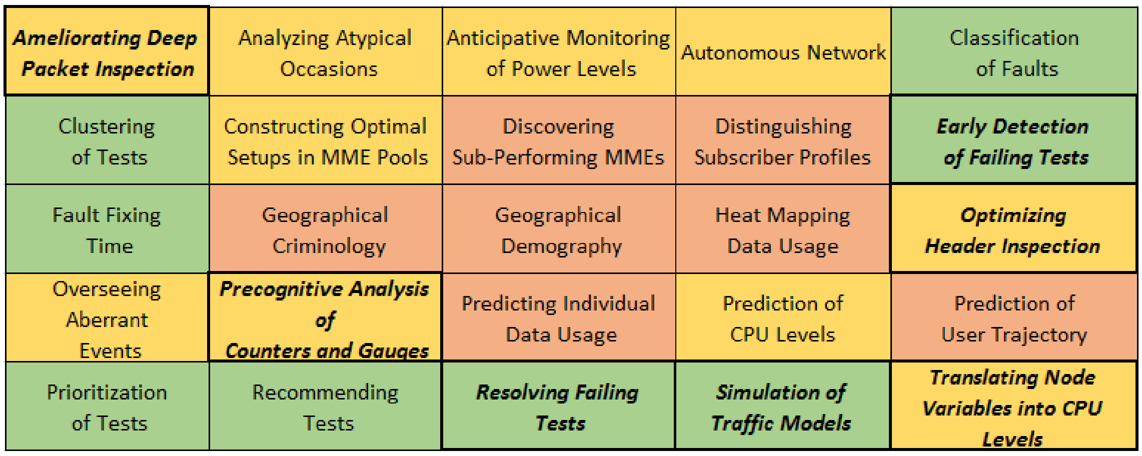
Description: A tool for computing the CPU levels induced in a node, based on its variable values.

Method: The tracking of counter and gauge values in a node, more specifically the incremented or decremented changes, can be useful for computing CPU levels in said node. Example nodes are the MME and PGW or SGW nodes. These many variables in a node have a complex relation to a corresponding CPU level, which current deterministic tools fall short of mapping accurately. However, a supervised method can learn these relations by taking the variable value changes as inputs and outputting a CPU level. Recommended machine learning algorithms are MLPs.

Expected Results: This tool should ideally be used to translate and predict CPU levels live, while the node is working. This could assist in determining whether to offload data payloading to another sibling node in the network.

Benefits: Typically, load balancing occurs in the EPG domain, since there are more of these nodes and they handle more data payloading. This model can therefore improve the decision making when offloading data pathways to other sibling nodes in the network.

4.3 Reviewing the Ideas



Ameliorating Deep Packet Inspection	Analyzing Atypical Occasions	Anticipative Monitoring of Power Levels	Autonomous Network	Classification of Faults
Clustering of Tests	Constructing Optimal Setups in MME Pools	Discovering Sub-Performing MMEs	Distinguishing Subscriber Profiles	***Early Detection of Failing Tests***
Fault Fixing Time	Geographical Criminology	Geographical Demography	Heat Mapping Data Usage	***Optimizing Header Inspection***
Overseeing Aberrant Events	***Precognitive Analysis of Counters and Gauges***	Predicting Individual Data Usage	Prediction of CPU Levels	Prediction of User Trajectory
Prioritization of Tests	Recommending Tests	***Resolving Failing Tests***	***Simulation of Traffic Models***	***Translating Node Variables into CPU Levels***

Figure 4.9: All the innovated ideas, the recommended ones are marked in bold.

After analyzing the innovated ideas in Section 4.2, the ideas listed below (also, see Figure 4.9) are deemed most feasible to execute on a practical level while also being profitable to prototype for a TSP.

Ameliorating Deep Packet Inspection

As mentioned in the description of the idea, TSPs wanting to achieve service identification of a package can do so by examining packets transmitted to and from the EPG node. However, a method that combines being computationally cheap and fully accurate is yet to have been introduced. Given this, the idea could be promising in terms of reducing resource usage when examining packets in the EPC network.

Early Detection of Failing Tests
With today's evolving market, it is becoming increasingly more important for companies to roll out updates and new features continuously. But, as with all software products, introducing changes to something that already works can always result in it breaking down instead. Of this reason, comprehensive testing of each new development is needed, while minimizing the time it takes to perform the screening. This is where the idea excels, as it decreases the time it takes to return the results from a given test.

Optimizing Header Inspection
As previously mentioned, there is a desire amongst TSPs to identify services used by a subscriber. There is however no method that is highly accurate, while being computationally cheap or lightning swift. On this basis, the idea should show promising potential of continuing the development of TSP networks.

Precognitive Analysis of Counters and Gauges
Monitoring network performance is vital if TSPs wish to maintain quality in communications transporting. Issues such as latency and downtime are ideally combated preemptively, rather than reactively. This idea should therefore be able to function well in tandem for TSPs whom proactively maintain the nodes of the EPC network.

Resolving Failing Tests
A source of time consumption in any work is typically manual labor. Identifying where or why a test case has failed is one such type of time consuming task. A tool that automates the process of problem identification can save significant time, even if it is only used as a recommendation rather than acted directly upon. This idea can therefore assist rookie test engineers in the learning process while streamlining workflow for more experienced ones.

Simulation of Traffic Models
When testing the introduction of a new feature, it is typically less costly for TSPs to do so in a virtual environment. In this virtual environment, everything needs to be simulated. This includes the behavior of a node, down to its internal mechanics. Potentially alleviating computational resources for emulating these nodes would be considered the main benefit from this idea.

Translating Node Variables into CPU Levels
A fairly obvious choice for machine learning application, this idea is also useful if TSPs are interested in having some level of precognition of CPU levels in a node. Ideally, this idea can assist in avoiding crashes and maintaining optimal performance in the network.

5 Conclusion

The telecommunications industry is one of the more recent industries to adopt the principles of machine learning, companies such as Ericsson are starting to realize how important and powerful a tool machine learning can be. The Evolved Packet Core network is one of the biggest and most complex systems that exist today, incorporating characteristics such as distribution, fault-tolerance and real-time communication. The network deals with massive amounts of data that is produced from the activities of its main and support nodes. Thus, it is befitting to employ machine learning in order to handle this data and improve the performance of the network at various scales.

In this thesis, we have studied and investigated the Evolved Packet Core network in order to establish areas of improvement by utilizing machine learning algorithms. The thesis began with a study of the various components of the EPC network, followed by an extensive literature review in order to explore existing ideas and solutions that involve the use of machine learning within the telecommunications industry. Rounding off was an ideation phase that involved close discussions with developers, designers, testers and managers within Ericsson in the form of either group presentations or face-to-face meetings in order to establish gaps within the EPC network.

The literature review resulted in 16 ideas and solutions that are applicable for implementation within the EPC network. The ideas were documented and categorized based on the problem they are addressing as well as the type of machine learning algorithm they use, see Section 4.1. The ideation phase resulted in 25 ideas for improving the EPC network, all of which are documented in Section 4.2. These ideas were divided into categories, and seven of them recommended in Section 4.3 for prototyping. Ericsson intends on proceeding with doing the recommended ideas as proof of concepts. Due to a combination of time limitations and the sensitivity of the data that the ideas deal with, it was decided that the prototyping is outside the scope of the thesis.

The thesis provides a detailed approach on the necessary steps that needs be taken when attempting to integrate new type of solutions, such as machine learning, into the complex environment of the EPC network. While the study and investigation of the thesis have taken place within Ericsson, the ideas that resulted from the thesis are applicable in any telecommunications-based system that shares similar characteristics to the nodes within the EPC network.

The next-generation network that will handle mobile data, also known as 5G, is currently being developed. The architecture is yet to be settled yet, leaving plenty of room for potential augmentation using machine learning with the goals of 5G being to further decrease latency while improving on already impressive bandwidth quantities. This increased bandwidth in the future will enable more devices to start transmitting data for the purposes of making human lives more convenient, anywhere and everywhere. The industry of IoT is expected to explode in the near future, with multiple applications available today already. As discussed in Section 4.1.7, these include transportation, healthcare, home utilities and manufacturing. We therefore conclude the thesis with that machine learning has come to stay in the telecommunications industry.

Bibliography

[3GP] 3GPP. 3GPP about 3gpp. `http://www.3gpp.org/about-3gpp`. Accessed: 2017-02-28.

[AAH$^+$09] T. Anagnostopoulos, C. Anagnostopoulos, S. Hadjiefthymiades, M. Kyriakakos, and A. Kalousis. Predicting the location of mobile users: A machine learning approach. In *Proceedings of the 2009 International Conference on Pervasive Services*, ICPS '09, pages 65–72, New York, NY, USA, 2009. ACM.

[ALMN99] D. E. Avison, F. Lau, M. D. Myers, and P. A. Nielsen. Action research. *Commun. ACM*, 42(1):94–97, Jan 1999.

[BHO16] A. M. Bosneag, S. Handurukande, and J. O'Sullivan. Automatic discovery of sub-optimal radio performance in lte ran networks. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 36–43, Apr 2016.

[BPS12] A. Booth, D. Papaioannou, and Anthea Sutton. *Systematic Approaches to a Successful Literature Review*. Sage Publications, 2012.

[BTA$^+$06] Laurent Bernaille, Renata Teixeira, Ismael Akodkenou, Augustin Soule, and Kave Salamatian. Traffic classification on the fly. *SIGCOMM Comput. Commun. Rev.*, 36(2):23–26, Apr 2006.

[CDGS07] Manuel Crotti, Maurizio Dusi, Francesco Gringoli, and Luca Salgarelli. Traffic classification through simple statistical fingerprinting. *SIGCOMM Computer Commununication Review*, 37(1):5–16, Jan 2007.

[CJB04] A. Collins, D. Joseph, and K. Bielaczyc. Design research: Theoretical and methodological issues. *Journal of the Learning Sciences*, 13(1):15–42, 2004.

[CMV13] X. Chen, F. Mériaux, and S. Valentin. Predicting a user's next cell with supervised learning based on channel states. In *2013 IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 36–40, Jun 2013.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[DVB15] B. Daroczy, P. Vaderna, and A. Benczur. Machine learning based session drop prediction in lte networks and its son aspects. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5, May 2015.

[DY14] L. Deng and D. Yu. Deep learning: Methods and applications. Technical report, NOW Publishers, May 2014.

[EHP06] P. A. Estévez, C. M. Held, and C. A. Perez. Subscription fraud prevention in telecommunications using fuzzy rules and neural networks. *Expert Systems with Applications*, 31(2):337–344, Aug 2006.

[Eri] Ericsson. Ericsson evolved packet core. `https://www.ericsson.com/ourportfolio/products/evolved-packet-core?nav=productcategory004`. Accessed: 2017-02-06.

[Fir] F. Firmin. 3GPP MCC the evolved packet core. `http://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core`. Accessed: 2017-02-28.

[FS11] H. Farvaresh and M. M. Sepehri. A data mining framework for detecting subscription fraud in telecommunication. *Engineering Applications of Artificial Intelligence*, 24(1):182–194, 2011.

[JWHT13] G. James, D. Written, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning with Applications in R.* Springer, 2013.

[LCB] Y. LeCun, C. Cortes, and C. J. C. Burges. MNIST the mnist database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`. Accessed: 2017-05-02.

[LJWC13] F. Liu, D. Janssens, G. Wets, and M. Cools. Annotating mobile phone location data with activity purposes using machine learning algorithms. *Expert Systems with Applications*, 40(8):3299–3311, 2013.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill Science/Engineering/Math, Mar 1997.

[MWG$^+$00] M. C. Mozer, R. Wolniewicz, D. B. Grimes, E. Johnson, and H. Kaushansky. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on Neural Networks*, 11(3):690–696, May 2000.

[QR89] J. R. Quinlan and R. L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80(3):227–248, 1989.

[Qui86] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, Mar 1986.

[Qui93] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[RKC10] R. Razavi, S. Klein, and H. Claussen. Self-optimization of capacity and coverage in lte networks using a fuzzy reinforcement learning approach. In *21st Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1865–1870, Sep 2010.

[Row12] Jennifer Rowley. Conducting research interviews. *Management Research Review*, 35(3/4):260–271, 2012.

[SHM+16] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, jan 2016.

[TL09] C. F. Tsai and Y. H. Lu. Customer churn prediction by hybrid neural networks. *Expert Systems with Applications*, 36(10):12547–12553, 2009.

[XG07] Dongshan Xing and Mark Girolami. Employing latent dirichlet allocation for fraud detection in telecommunications. *Pattern Recognition Letters*, 28(13):1727–1734, 2007.

[YWD04] L. Yan, R. H. Wolniewicz, and R. Dodier. Predicting customer behavior in telecommunications. *IEEE Intelligent Systems*, 19(2):50–58, Mar 2004.

[ZISAD14] A. Zoha, A. Imran, A. Saeed, and A. Abu-Dayya. A machine learning framework for detection of sleeping cells in lte network. In *1st Machine Learning and Data Analysis Symposium*, Jan 2014.