?Generic Datasets, Beamforming Vectors Prediction of 5G Celleular Networks

CAPSTONE REPORT By Manjit Singh, singhm1@sunypoly.edu
Under Supervision of Dr. Hisham A. Kholidy, hisham.kholidy@sunypoly.edu
State University of New York Polytechnic Institute
Utica, NY 13502, UNITED STATES

Abstract

The early stages of 5G evolution revolves around delivering higher data speeds, latency improvements and the functional redesign of mobile networks to enable greater agility, efficiency and openness. The millimeter-wave (mmWave) massive multiple-input-multiple-output (massive MIMO) system is one of the dominant technology that consistently features in the list of the 5G enablers and opens up new frontiers of services and applications for next-generation 5G cellular networks. The mmWave massive MIMO technology shows potentials to significantly raise user throughput, enhances spectral and energy efficiencies and increases the capacity of mobile networks using the joint capabilities of the huge available bandwidth in the mmWave frequency bands and high multiplexing gains achievable with massive antenna arrays.

In this report, we present the preliminary outcomes of research on mmWave massive MIMO (as research on this subject is still in the exploratory phase) and study two papers related to the Millimeter Wave (mmwave) and massive MIMO for next-gen 5G wireless systems. We focus on how a generic dataset uses accurate real-world measurements using ray tracing data and how machine learning/Deep learning can find correlations for better beam prediction vectors through this ray tracing data. We also study a generated deep learning model to be trained using TensorFlow and Google Collaboratory.

Keywords*: massive MIMO, Millimeter Wave, 5G Networks, Supervised Regression, convolutional Neural Network, Machine Learning, ?Generic Datasets, Beamforming Vectors*

1. Introduction

Millimeter wave and massive MIMO systems is considered next-gen wireless technology (Alkhateeb, {2019}). This is due to the high data rate and the large bandwidth (Alkhateeb, {2019}). These characteristics makes mmWave and massive MIMO systems an ideal candidate in next-gen 5G networks, however these systems still have a few restraints and challenges that can be overcome according to the original authors (Alkhateeb, {2019}). We study two papers related to the Millimeter Wave (mmwave) and massive MIMO. The 1st paper on Deep MIMO discusses how a generic dataset is needed and how it can further push research in these systems due to the increase in Machine Learning (ML) applications in this system. The paper goes over how the dataset uses accurate real-world measurements using ray tracing data and how machine learning/Deep learning can find correlations for better beam prediction vectors through this ray tracing data. The 2nd paper discusses exactly how their solution achieves these accurate beamforming predictions by detailing the innerworkings and overall approach that leverages the ray tracing data and how it compares to their baseline solution, which they consider to be more exhaustive and inefficient compared to their proposed coordinated beamforming solution. The next few sections of this report discuss the systems and tools used to produce the steps outlined in the two papers being reviewed, such as the TensorFlow system, what a deep learning model is and classification techniques etc. and provide more detail as to how the approach of the literature

reviews are being utilized. Followed by outline steps to reproduce these results and then end with an evaluation of the deep learning model used in this project, and finally a comparison of the results to the authors work.

The challenges of how to get around blockage, which could potentially affect coverage and reliability, so the environment topology needs to be taken into consideration and when a user walks through the environment how Base Stations (BSs) handle the process of passing that user onto the next base station needs to be done with little delay (A. Alkhateeb, 2018). This is where Machine Learning capabilities are being leveraged (Alkhateeb, {2019}). Machine learning applications are on the rise in many areas, due to how they can handle large amounts of data, automate and fine tune themselves through hidden layers (Gupta, 2020), as well as figuring out optimization problems through correlations in data and are used to find gaps that could be identified, minimized and leveraged to create more robust technologies (Alkhateeb, {2019}). The reason why machine learning, or more specifically Deep learning is used is so that they can make predictions for us and the more data that is input the more accurate they can become (Donges, 2017). This is beneficial for making predictions since these machines can find relationships between input data to make these accurate predictions, like predicting beamforming vectors (A. Alkhateeb, 2018). So, it has been stated that instead of paying for more advanced technologies to achieve high data rates, instead it's easier to take advantage of the technology that already exists and simply make them more efficient in how they approach the goal of the "highest achievable rate" (A. Alkhateeb, 2018).

This report will go into reproducing the authors work of their coordinated beamforming solution. This solution is classified as a Supervised Regression NN with a generic dataset. We study how much their solution is efficient and why, as well as how other architecture types such as a Convolutional NN have further improved upon their Coordinated Beamforming solution implying that other architectures types should be researched (A. Alkhateeb, 2018).

2. Challenges and constraints.

The initial start of project proved challenging as there was a very steep learning curve. There were many topics that needed to be reviewed before I had a proper understanding of where to begin this project. I had to become accustomed to TensorFlow a framework where I would later train the regression-based model. TensorFlow, provided by google has the resource capabilities necessary to train and test the model with the generic dataset. I also needed to setup a VM with high resource capabilities, the answer was to use Google's cloud VM compute services to meet the resource requirements to generate the dataset and model algorithm .I also learned to work inside MATLAB and become comfortable analyzing the code to understand exactly what the source codes were doing. It was necessary to learn the source codes of the dataset and model algorithm, due to how the channel parameters and ray tracing data defined the entire model and dataset, so it is necessary to understand the adjustments being made as the entire generation process revolves around the parameters being set. The other challenge was once the parameters and model were generated

using the "01" scenario ray tracing data, which is data that has real-world measurements of the environment topology you are simulating. The resultant model created around that measurement data, and the entire dataset needs to be saved inside MATLAB in the proper MATLAB format, known as the V.7 format. If the wrong format type was used, such as the V7.3 format the python code provided with the source codes of this Beamforming solution to compute the data will run into errors inside TensorFlow when training and testing of the model was begun. Lastly the final issue I came across was how to import the results of the training and testing back into my cloud VM in order to run the figures and plot the final graph for comparison purposes. The solution was to move the files onto Google Drive Desktop instead of downloading them off of google drive, which improperly splits the files due to download size limitations. It is simply easier to sync your drive to your desktop and move the files that way, however I still needed to move those files from my local machine desktop into my cloud VM. Since google drive desktop does not have support in Linux, I moved the result files from my google desktop sync onto my local machine's MATLAB drive, which I had already downloaded and set in my VM so the files would sync there for me to plot the graphs inside MATLAB.

3. Literature Review.

3.1 The first paper: *DeepMIMO: A Generic Deep Learning Dataset for Millimeter Wave and Massive MIMO Applications"* (Alkhateeb, A Generic Deep Learning Dataset, {2019}). A

Literature Review Study

As machine learning applications are being leveraged in the next-gen wireless, known as mmWave and massive MIMO (A. Alkhateeb, 2018). The authors propose why advancements in this research require a common dataset to test algorithms and compare different solutions (Alkhateeb, {2019}). The comparison of algorithms through a common dataset can be used to benchmark and reproduce results through adjustments of channel parameters and measurement data by other researchers (Alkhateeb, {2019}). There is a large amount of mathematics behind these algorithms and can be simplified through channel parameters and ray tracing data (Alkhateeb, {2019}). However, it has been stated that since mmWave and massive MIMO use a large amount of antenna arrays it becomes difficult to coordinate a mobile user reliably with little complexity (Alkhateeb, {2019}). This requires long training time and computation time and is a drawback of using large channel matrices, but since the authors of the generic dataset state that these matrices hold some function of the environment geometry and surrounding materials, as well as the number of base stations and their locations etc. a machine learning or Deep Learning tool can be used to optimize and leverage these functions to predict the best beamforming vectors (Alkhateeb, {2019}). The authors state how machine learning approaches in mmWave and massive MIMO are on the rise, they also proposed a common dataset in order to compare and reproduce results of other researchers to further improve these technologies (Alkhateeb, {2019}). This can help streamline research in this field. They describe the promising capabilities of these next-gen wireless systems, but also point out the complexity of using these technologies and the latency issues associated with them (Alkhateeb, {2019}). They propose that since ray tracing data holds key functions of the

environment a deep learning approach can correlate relationships between the mobile user and the environment to predict optimal vectors (A. Alkhateeb, 2018). The goal of this literature review is to analyze why a common dataset is needed to further research in mmWave and massive MIMO systems.

A Need for a Generic Dataset.

The accurate measurement and deployment of machine learning tools is very important. It is also important that researchers can compare and contrast between solutions easily, as this promotes research and simplifies the complexity already associated with the large channel matrices in mmWave and massive MIMO (Alkhateeb, {2019}). Due to the large amount of data available, its intuitive that researchers would opt for machine learning applications in the next-gen wireless systems, as sifting through this data is largely automated though deep learning (Gupta, 2020). The benefit of a common dataset helps to create a better understanding among researchers to propose better alternatives amongst one another and simplify the mathematics behind their algorithms through adjustments of simple channel parameters and the ray tracing scenario (Alkhateeb, {2019}). In short, the common dataset allows researchers to compare different algorithms to test and benchmark other researcher's approaches to see how they compare to one another (Alkhateeb, {2019}). This helps to optimize the capabilities of the next-gen wireless and overcome the issues that also come with these promising technologies and work towards the goal of making them more efficient in predicting the best beamforming vector for reliability of coverage to mobile-users (Alkhateeb, {2019}).

Conclusion.

In conclusion based upon the evidence given by the authors, it makes sense that as machine learning and deep learning tools can leverage the large amount of data available with the mmWave and massive MIMO systems. The use of channel parameters and ray tracing data to accurately simulate the environment and define the dataset and algorithm simplifies and ensures that the results obtained from the deep learning output are reliable (Alkhateeb, {2019}). As research in the next-gen wireless domain rises its ideal for researchers to have a common dataset to compare and test each other's algorithms, or even propose other Neural network model architectures with the key goal of optimizing and improving the issues attached with the large channel matrices in these systems for more accurate beamforming predictions using deep learning tools and streamline research.

3.2 The second paper: "Deep Learning Coordinated Beamforming for Highly-Mobile Millimeter Wave Systems" (A. Alkhateeb, 2018).

Introduction.

In order to support a mobile user as they move from one BS to another and ensure reliability in mmWave and massive MIMO there are some challenges that need to be overcome to ensure reliability from signal blockage and hand-off between base stations with little delay (A. Alkhateeb, 2018). It has been found that the approach of predicting an optimal beamforming vector with deep learning can get around these issues, since the environment topology and the materials in the

surrounding largely impact how well a user can be provided with reliable coverage (A. Alkhateeb, 2018). Since it is stated that environment topology is a key factor in how well a prediction can be made, GPS data has been considered by the authors for analyzing the user location, but also state has a few drawbacks (A. Alkhateeb, 2018). The goal of this literature review is to compare how their proposal of an uplink training pilot sequence and omni/omni-quasi beam patterns compare to GPS data when it comes to accurately predict beamforming vectors (A. Alkhateeb, 2018).

Main Body.

It has been stated that GPS data can provide user location information, however there are drawbacks as it does not carry key signatures of the nature of the environment, which can impact the coverage of the user and cause delays (Alkhateeb, {2019}). A user's whereabouts, (whether indoors, or outdoors etc.) within the environment is important, as materials, (like concrete, wood etc.) can impact how well a signal propagates through the environment (A. Alkhateeb, 2018). According to the authors of the paper, a better solution would be to use something called the uplink training pilot signature with omni/quasi-omni beam patterns, as it better captures the nature of the environment (Alkhateeb, {2019}). It is suggested that in order for accurate beam prediction more information needs to be known than just the location of the user, since the nature of their environment can impact how well the signal is received (A. Alkhateeb, 2018). Since the goal is to provide good coverage with little to no delay GPS data is not well received in Line of Sight or Non-Line of sight situations, whereas the omni-quasi beam patterns trained inside a Neural Network model can adapt to its environment as proposed by the authors in what they called the Coordinated Beamforming solution (Alkhateeb, {2019}). It makes sense to use more than just the location of the user since signals need to overcome attenuation, absorption, diffraction etc (A. Alkhateeb, 2018). the topology information held within the omni beams are stated to be better suited for prediction purposes rather than just the location alone (Alkhateeb, {2019}). The information contained in these sequences can be trained by a NN to adapt to the environment since neural networks can be used to find correlation between a user's location and the nature of their environment (A. Alkhateeb, 2018). In short, the authors state that GPS data alone is not enough to predict vectors if the information contained with it is just a location of the user (A. Alkhateeb, 2018). The materials and geometry information and the location would be better suited for leverage by deep learning tools, which the omni beams and uplink-pilot do contain whereas GPS data only supplies user location, which in the case of beamforming vectors is not sufficient (A. Alkhateeb, 2018).

Since the goal of the authors is to reach the highest achievable rate, efficiency was necessary (A. Alkhateeb, 2018). So, their goal was to compare how well their solution stood compared to a more exhaustive approach in what they called the baseline beamforming solution (A. Alkhateeb, 2018). This solution searches for every possible combination between the uplink training pilot and the omni beams (A. Alkhateeb, 2018). While such a method according to them can achieve their goal on some occasions, it was stated that it does not address their goal of efficiency compared to their proposed Coordinated Beamforming solution, which trains the data and then switches to a

prediction phase and made accurate predictions based only on the omni beams, whereas their baseline solution needed more training time and computation time and is a more exhaustive approach (Alkhateeb, {2019}).

Conclusion.

In conclusion the goal of the author's was to reach efficiency using deep learning tools. The authors provide why GPS data is not sufficient amount of information compared to their proposal of the uplink training pilot and omni code book beams. Since key information of the topology and geometry and materials is available within these signatures, and provides better and more accurate predictions through these combinations of sequences (A. Alkhateeb, 2018). In short, their baseline solution pales in comparison to their proposed coordinated solution since the deep learning model can continue to predict and adapt once it has been trained with these sequences as opposed to searching for all possible combinations of possible vectors (Alkhateeb, {2019}).

4. Main Contribution of this report.

The goal of this section is to go over the systems and tools used to conduct this project. The goal of this project was to reproduce the works within literature reviews above using the "01" scenario ray tracing software and adjusting the channel parameters according to this scenario to compare the output to that of the authors, such as the baseline solution, coordinated beam solution in regards to what they call the genie-aided solution, which already knows the best beamforming vectors (A. Alkhateeb, 2018). Some of the tools used for example such as TensorFlow, Deep learning and classification techniques will be reviewed in this section.

In this section, we will outline concepts, tools and systems that need to be understood, such as what TensorFlow, and deep learning is and its benefits over traditional learning approaches, as well as some classification techniques used in ML in order to classify outputs for comparison and evaluation purposes in the later sections.

Deep Learning.

Deep learning solutions have gained significant momentum due to the amount of data we have accumulated (Donges, 2017). The solutions they provide take advantage of this large amount of data whereas traditional algorithms do not (Donges, 2017). This may be due to their approach in how they handle data, which can automate and fine tune itself to match a prediction as closely as possible (Gupta, 2020). The traditional algorithms, such as logistic regression and Support Vector Machines (SVMs) do increase in accuracy like the Deep learning algorithms, but begins to fall as you give them more data (Donges, 2017) , A deep learning algorithms advantage is its ability to adapt and fix itself whereas in deep learning it does not, however it is still possible to cause overfitting in the case of deep learning where you give them too much data, for which there are ways to alleviate any distortions in output through different methods, such as the dropout layers, like the one used in the authors proposed coordinated solution (A. Alkhateeb, 2018). For this

project deep learning was used to find correlations between the uplink pilot and the omni beams, after the learning phase of this data (A. Alkhateeb, 2018). The model is able to find the relationship between the variables, (or uplink and omni/codebook beams) and use that information to predict the most optimum vectors to provide reliable coverage to a mobile-user as they move from one BS to another, and to compare that prediction to the actual prediction (genie-aided solution) (A. Alkhateeb, 2018). The goal of using deep learning within this project was to approach what the authors called their genie-aided solution, which is what the deep learning model tries to reach as closely as possible and continuous to calculate constantly, which is quality of a regression based model used by the authors of the coordinated solution (A. Alkhateeb, 2018).

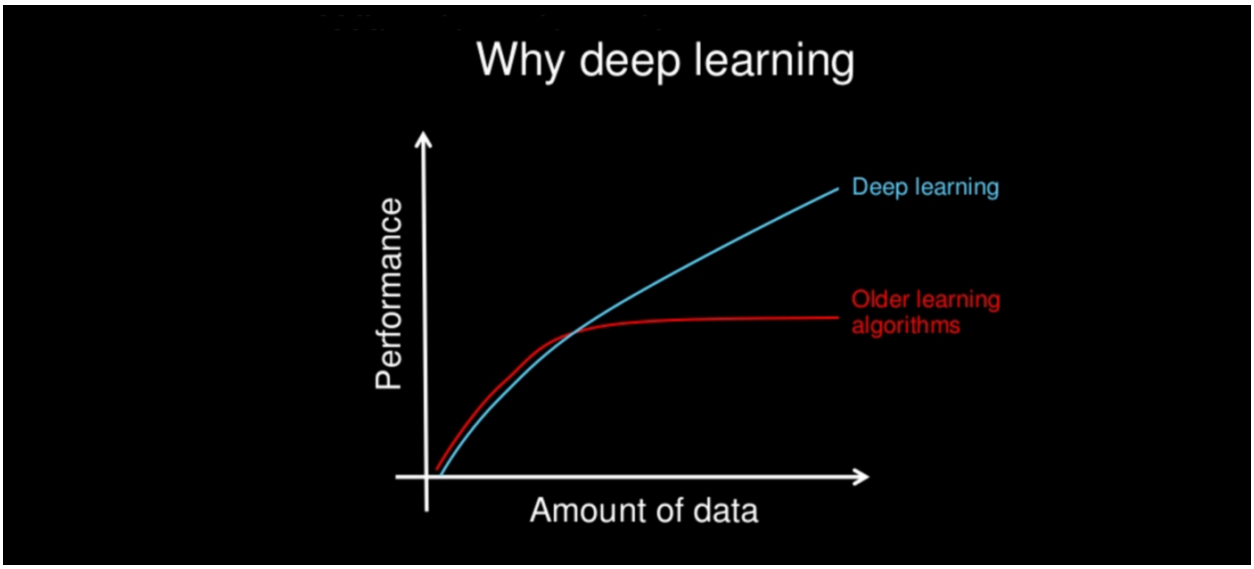


Image source: https://machinelearning-blog.com/2017/11/03/erster-blogbeitrag/

This image illustrates the performance benefits of a deep learning model over traditional algorithms. As the amount of input data is given to the deep learning model its accuracy continuous to increase whereas the traditional algorithms decline (Donges, 2017).

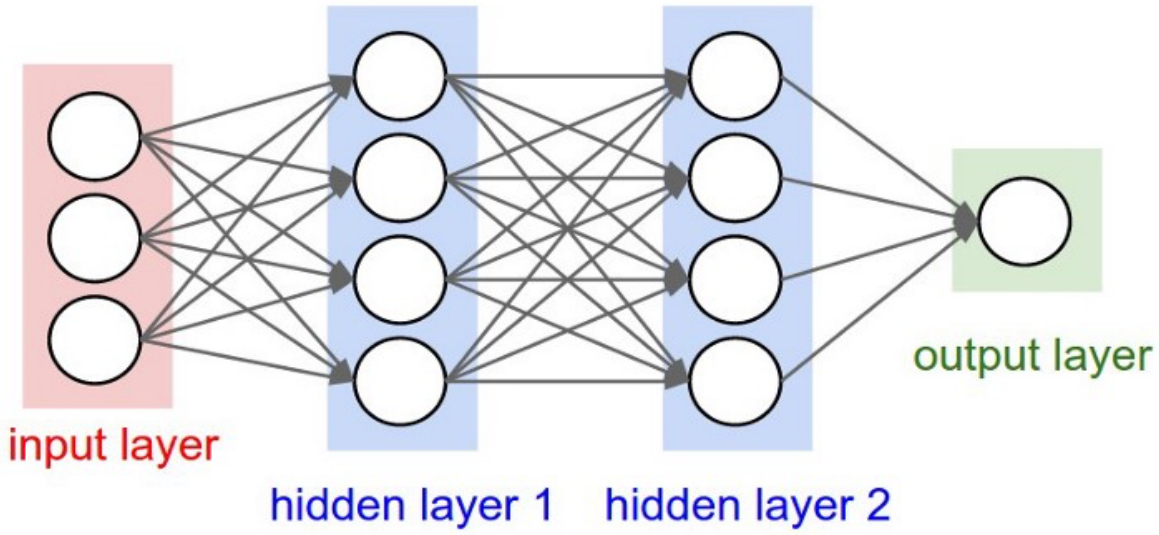


Image source: https://medium.com/@rajatgupta310198/getting-started-with-neural-network-for-regression-and-tensorflow-58ad3bd75223

This image illustrates a simple example of what a neural network can look like. The input is taken and there can be any number of hidden layers in between (Gupta, 2020). These hidden layers each pass and fine tune the information as it passes through them and give you a resultant output (Gupta, 2020). It has been stated that in a fully-connected model each input layer produces an output, which is taken as an input for the next layer (A. Alkhateeb, 2018).

More specifically for this project, a Supervised Regression based NN model was used. More detail of what Supervised learning and regression-based models are will be detailed below based on a diagram used in the authors papers.

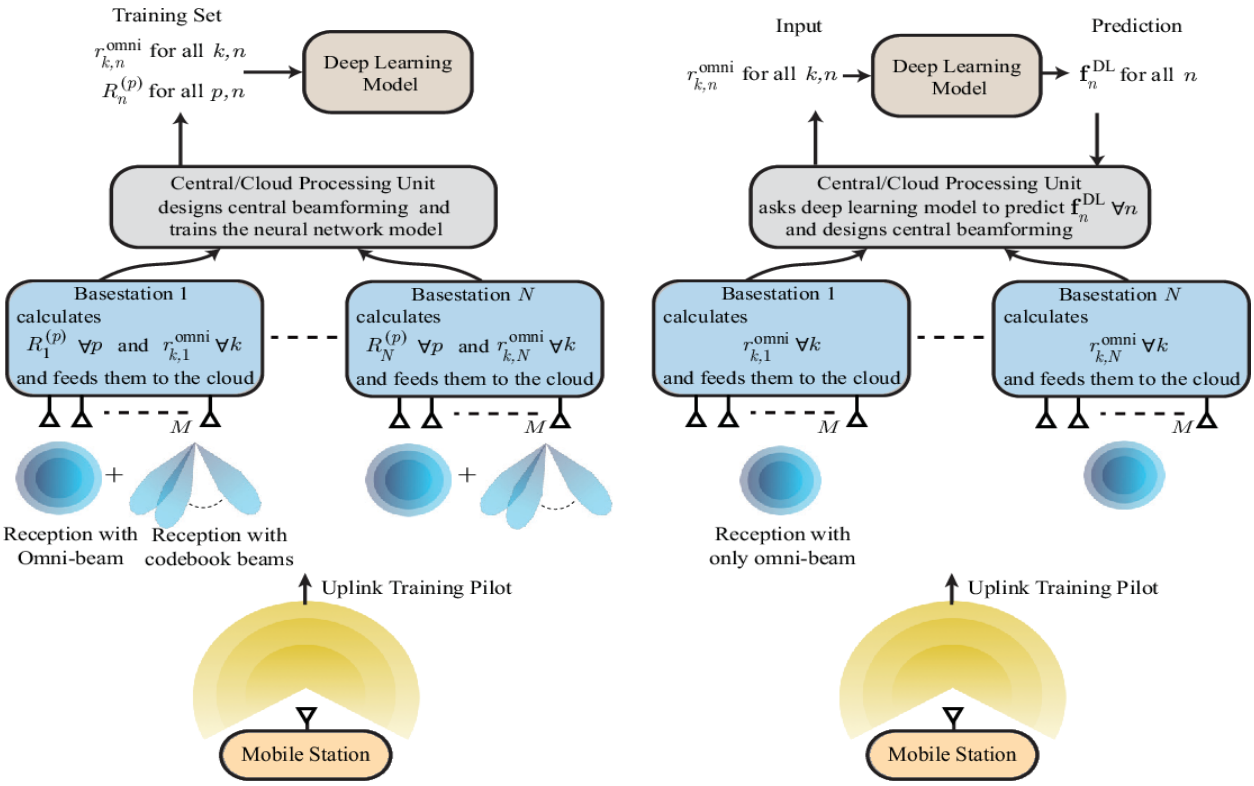


Image Source: https://www.semanticscholar.org/paper/Deep-Learning-Coordinated-Beamforming-for-Wave-Alkhateeb-Alex/cc779ba71eabdf68944051eea852a24eaf79092c/figure/2

This diagram illustrates the Supervised regression-based model used in the authors paper. This model has two phases. The 1st phase is the learning phase, the DL model will leverage the signals that are received (uplink pilot and omni/code book beams) and will be trained for all possible combinations of vectors, called the baseline solution (A. Alkhateeb, 2018). When the affective achievable rate is greater than the baseline solution it will switch over to the 2nd phase, known as the prediction where it will only use the omni beams to predict optimal vectors (A. Alkhateeb, 2018). It has been stated that the model will only switch to the 2nd phase only when the highest

achievable rate can be reached and will switch back periodically since the output values are continuous and the cloud processor continues to constantly calculate (A. Alkhateeb, 2018). This is why this model is called a regression model since the output value is constantly being calculated by a cloud processor in the case of this solution model and this helps to adapt to any environment once it has been trained, which is an advantage over the baseline solution (A. Alkhateeb, 2018). In machine learning sometimes giving models too much data can result in inaccuracies in output (A. Alkhateeb, 2018). This is called overfitting and for this model something called a rectifier linear units (ReLU) activation is used at each layer and every one of these layers has a drop-out layer to avoid overfitting the NN (A. Alkhateeb, 2018). This way the model stops training on its own (A. Alkhateeb, 2018).

TensorFlow.

TensorFlow is a large framework that allows beginners and experts to create machine learning models on multiple devices (Yegulalp, 2018). This open-source library that is python friendly is used for numerical computation that makes machine learning faster and easier (Yegulalp, 2018). Since machine learning is complex TensorFlow makes it easier to obtain data, train the models and serve the prediction results (Yegulalp, 2018). For this project once the dataset and algorithm were generated, they needed to be trained inside TensorFlow in order to give us the predictions, which were the figures that were generated and plotted onto a graph. The resultant graph will be displayed in the ***Procedure*** *sub-section* later down below. The python code that did all the computation was called DL_python_code.py (A. Alkhateeb, 2018).

Classification Techniques.

In Neural networks (NN) there are three types of learning, Supervised, Unsupervised, and Reinforcement (Knocklein, 2019). For this project a Supervised learning method was used, which is also the simplest one of the three (Knocklein, 2019). In this type of learning your input data is labelled and the output is generalized, so that in it can be applied to unlabeled cases (Knocklein, 2019). For example, the uplink training sequence and the omni beams are the labelled data that the NN will learn and attempt to generalize for future unlabeled outputs (A. Alkhateeb, 2018). For this project the genie-aided solution are the predictions we want to approach and the output data from the learning phase tries to generalize the input it receives to match as close as it can to the predictions we would like to see (A. Alkhateeb, 2018). So, basically the labelled inputs are trained to learn, so that the model can generalize what it learns, so that it can apply that to unlabeled data and try to match the genie-aided solution stated in the author's paper (A. Alkhateeb, 2018). Unsupervised learning is where the data does not have labels and deduces based on some rules and mathematical functions on the data it is being given, this also applies to the output (Knocklein, 2019). This type of classification is used for clustering and not predictions (Knocklein, 2019). This type of learning was not applied to this project as predictions of the beam forming vectors were being made and not clustering of any data.

Regression NN.

In these models, labelled data is required to have continuous values like in the two phased model, where it periodically switches back and forth between learning and predicting (A. Alkhateeb,

2018). This model tries to minimize the difference between what the actual value is and what the predicted value came out to be and the results are continuously calculated (Malik, 2018).The benefit of this model is that in terms of the most optimal beamforming vectors this model can list the 1st, 2nd, 3rd best vectors and so on (A. Alkhateeb, 2018), These vectors would be listed based on strongest transmit power, and the results of these models are continuously calculated, as it switches back and forth between learning and predicting periodically (A. Alkhateeb, 2018). The next sub-section will outline the steps taken for this project.

Procedure:

- Step1: setup a cloud VM, The VM was created using Google Cloud services in order to satisfy the resource requirements of being able to run the Dataset generation code as well as generating the Beamforming algorithm.
- Step2: Download VNC Client on your host machine. This is in order to have a graphical desktop display instead of a terminal in the cloud VM. VNC provides a graphical remote desktop. Download the VNC server version on your cloud VM through the terminal and turn it on using the ***command****: tightvncserver.*
- Step3: Once your cloud VM is setup and you are able to remotely connect using VNC client, which is to be installed on your host machine, Install MATLAB on your cloud VM, you will need a license in order to be able to use MATLAB and its tools.
- Step4: Download the files necessary to create the dataset such as the DeepMIMO master code folder and the 01-scenario available on the authors of Deepmimo(.)net as well as the DL_Beamforming algorithm folder.
- Step5: Unzip both the DeepMimo master code and DL beamforming folders and place the "01" scenario folder inside the "Ray Tracing" folder of the Master code folder directory. Place the DL beamforming folder inside the root of the Deep MIMO code master folder.
- Step6: Run MATLAB and set the MATLAB path to be able to "see" the DeepMimo master code folder. Be sure to right-click and add all the folders and sub folders to the MATLAB path
- Step7: Set the following Channel parameters stated by the authors of Deepmimo for the "01" scenario which are:

DeepMIMO Dataset Parameter	Value
Active BSs	3, 4, 5, 6
Active users	From row R1000 to row R1300
Number of BS Antennas	$M_x = 1, M_y = 32, M_z = 8$
Antenna spacing	d=0.5
System bandwidth	B=0.5 GHz
Number of OFDM subcarriers	1024
OFDM sampling factor	1
OFDM limit	64
Number of paths	5

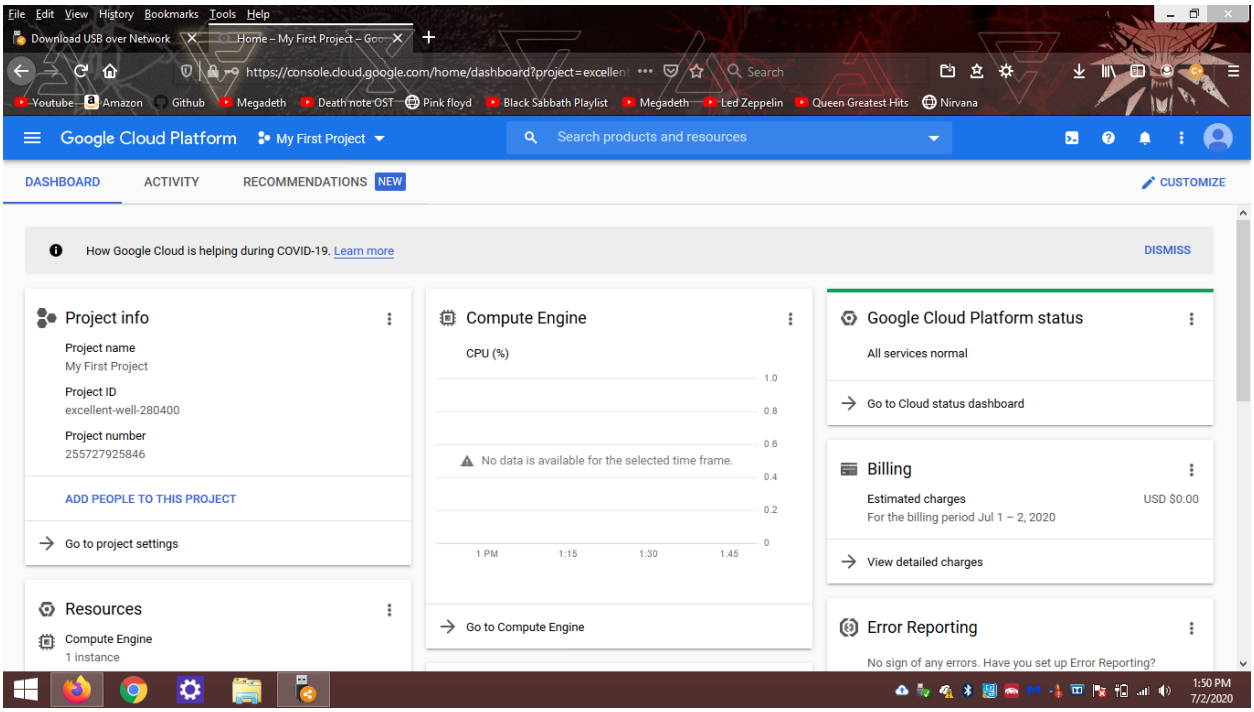
Table source: https://www.semanticscholar.org/paper/DeepMIMO%3A-A-Generic-Deep-Learning-Dataset-for-Wave-Alkhateeb/d1d8980d04d411c314910d1926f3bbaac46c2197#extracted

Table source: https://www.semanticscholar.org/paper/DeepMIMO%3A-A-Generic-Deep-Learning-Dataset-for-Wave-Alkhateeb/d1d8980d04d411c314910d1926f3bbaac46c2197/figure/1

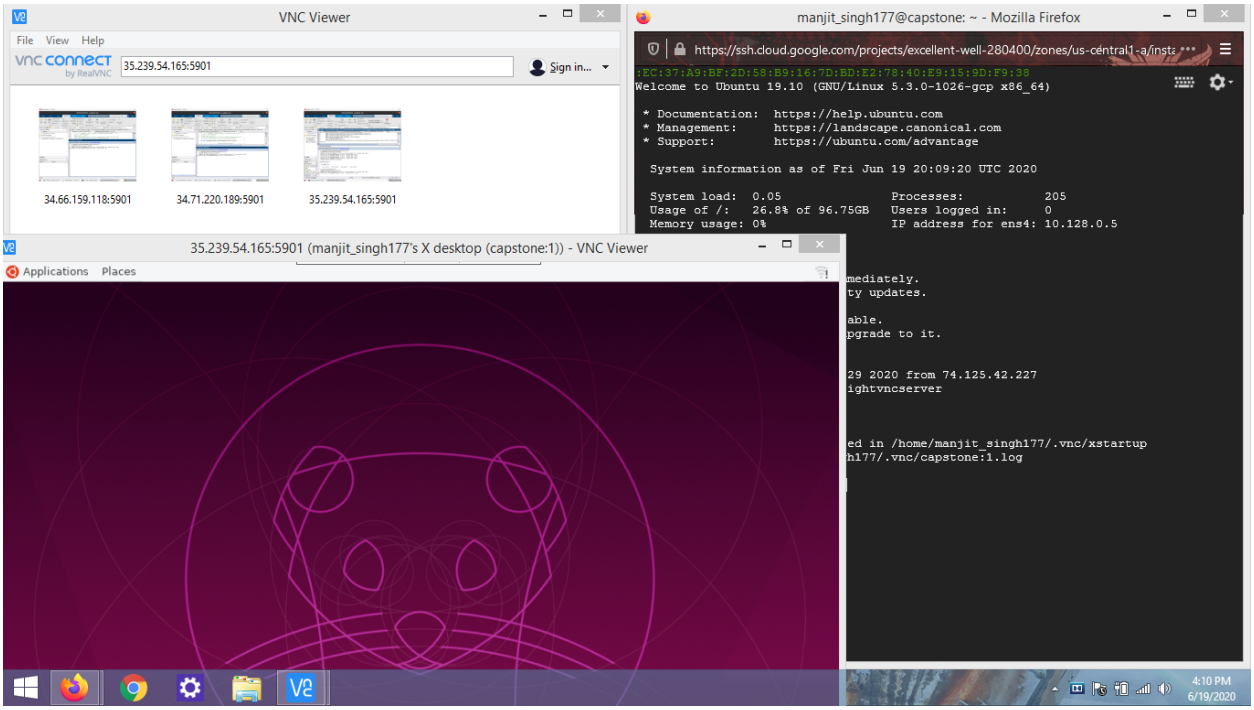
Here is a screenshot of the "01" scenario provided by Wireless InSite by Remcom. This was the scenario being used in this capstone project.

- Step8: Once the channel parameters have been set run the Dataset generation code and wait for it to complete and save a file called dataset.mat
- Step9: Once you have created the dataset run the DL model beamforming.mat code to generate the DL model. You should end up with the input/output.mat for the Deep learning model.
- Step10: Now it is time to use TensorFlow framework inside google colab to build, train and test your model. This will require at least Python version 3.6, Keras library. Run the DL model python code named DL_model_python_code.py to begin testing.
- Step11: Once the testing has finished you will end up with some result files inside your DLCB_code_output folder. You will now need to move these files onto your cloud VM for use in MATLAB and then convert your result files into ". mat " format to be read by the figure_generator.m source code file. Run the figure_generator.m code, be sure the DLCB_code_output folder has the result files inside and make sure the directory is visible to MATLAB before you run.
- Step 12: Once the figure generator has run and completed. MATLAB will plot your graph and you will end up with your results.

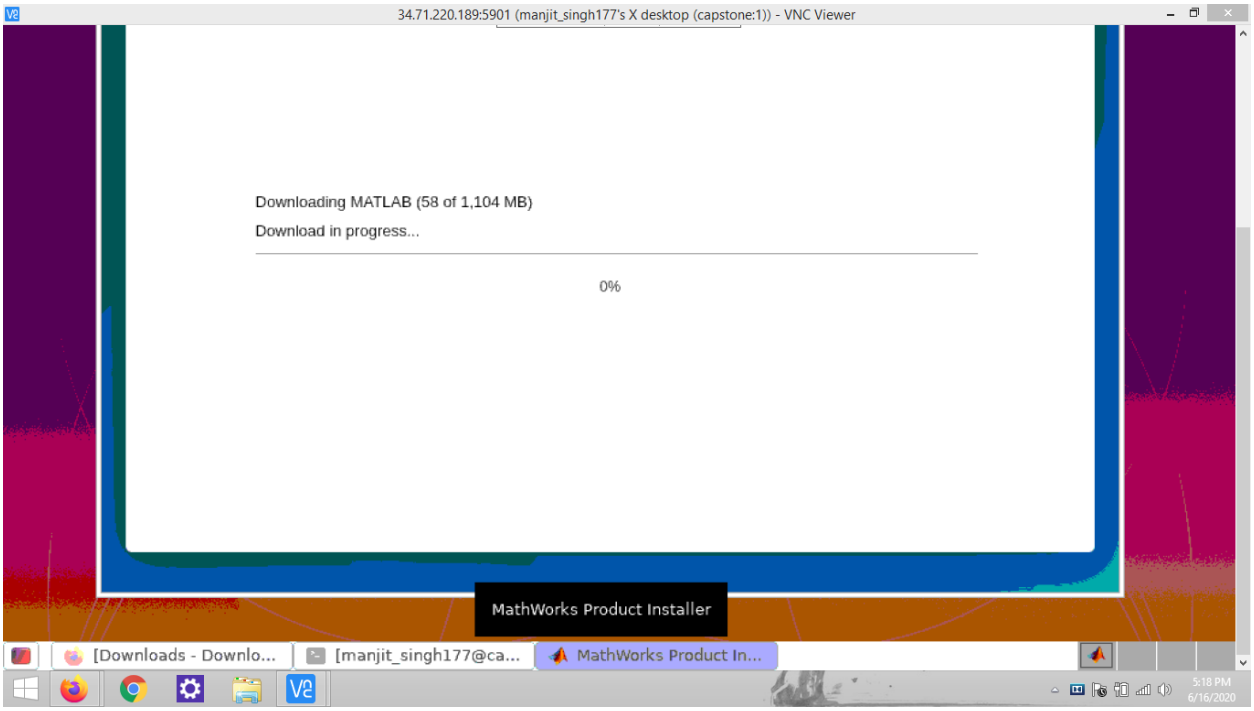
The steps aboved will be now illustrated using screenshots:



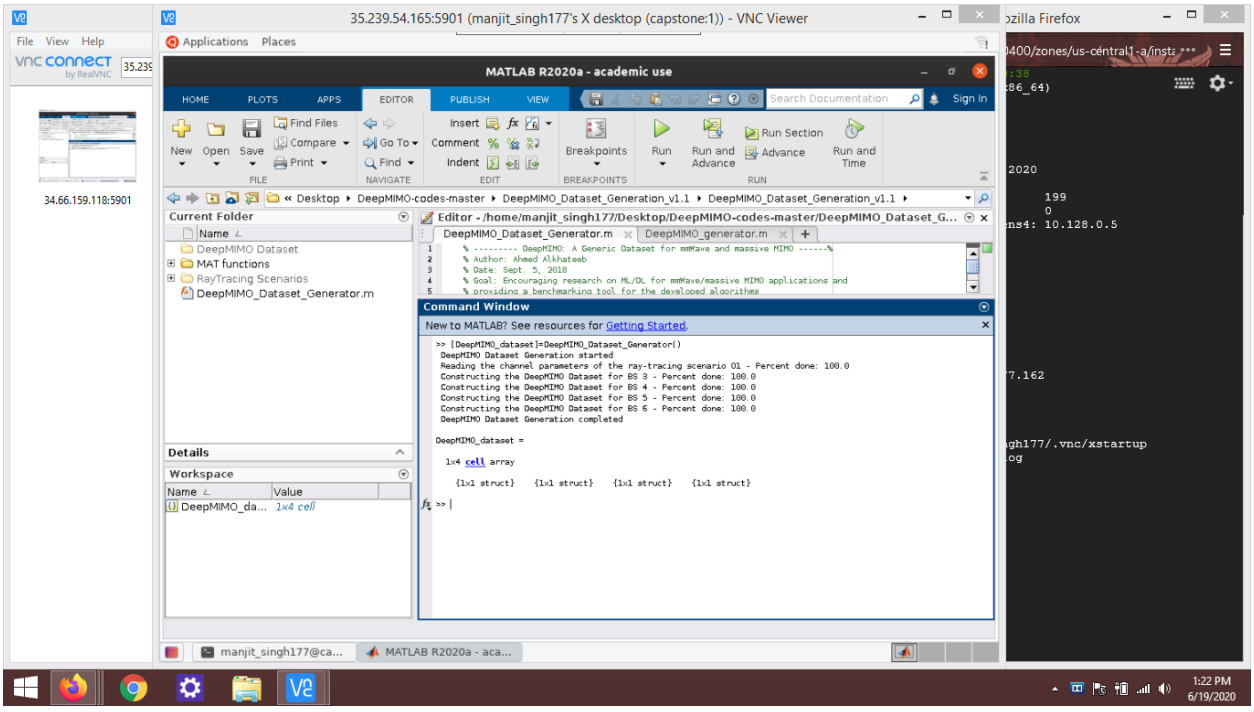
Setup a cloud VM to satisfy resources necessary to generate dataset and beamforming algorithm. Here Google cloud was used to get the resources necessary for dataset and algorithm generation.



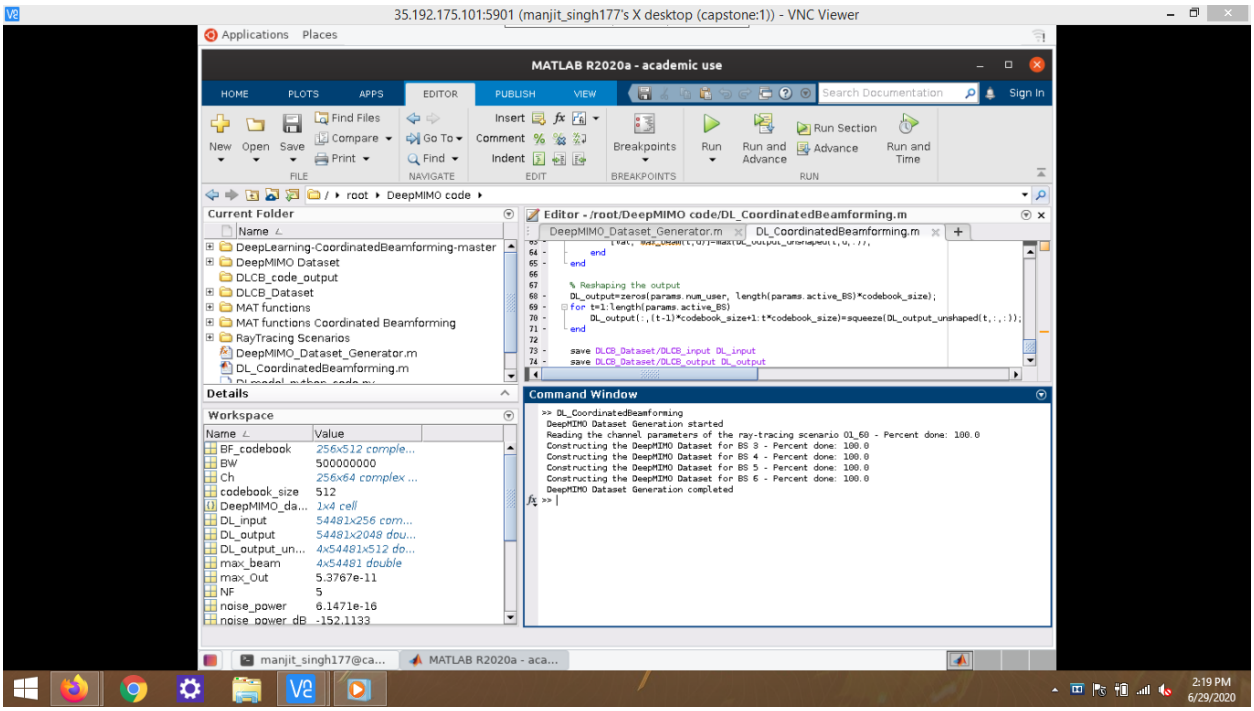
Setup VNC client on your host machine and VNC server on your cloud VM, so you will have a graphical remote desktop as opposed to just a terminal to work with.



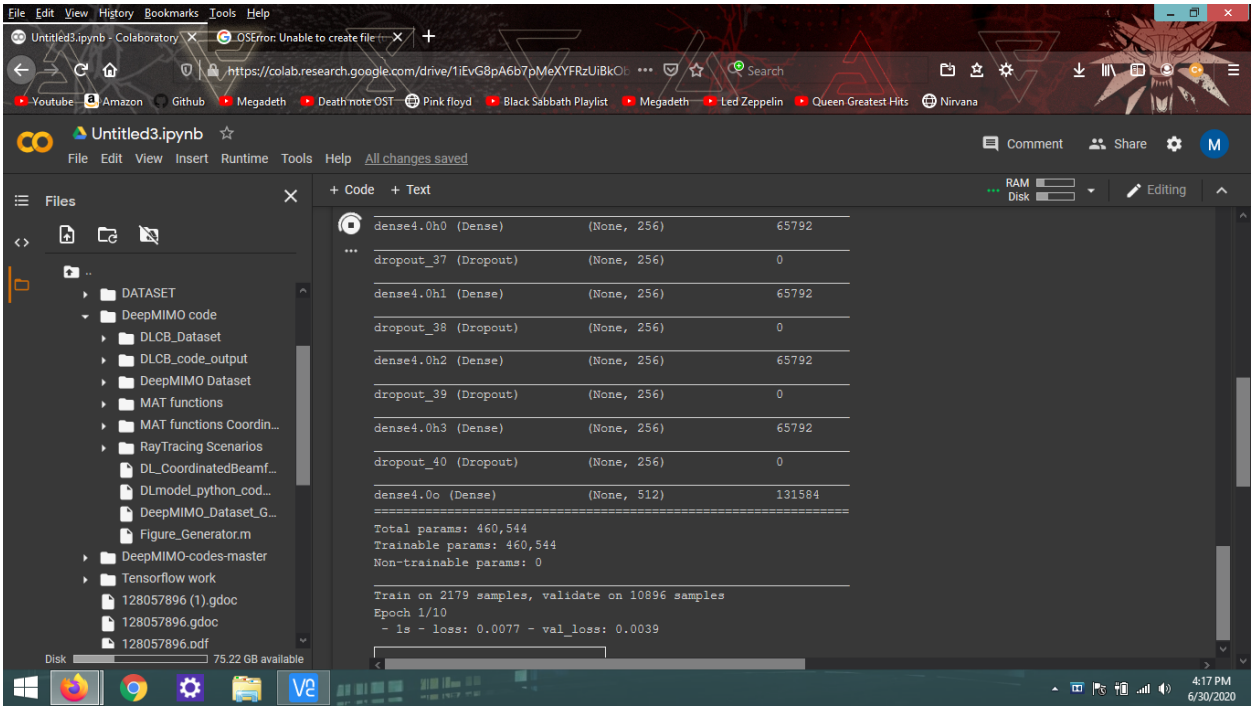
Download and install MATLAB. A license is required to use these tools.



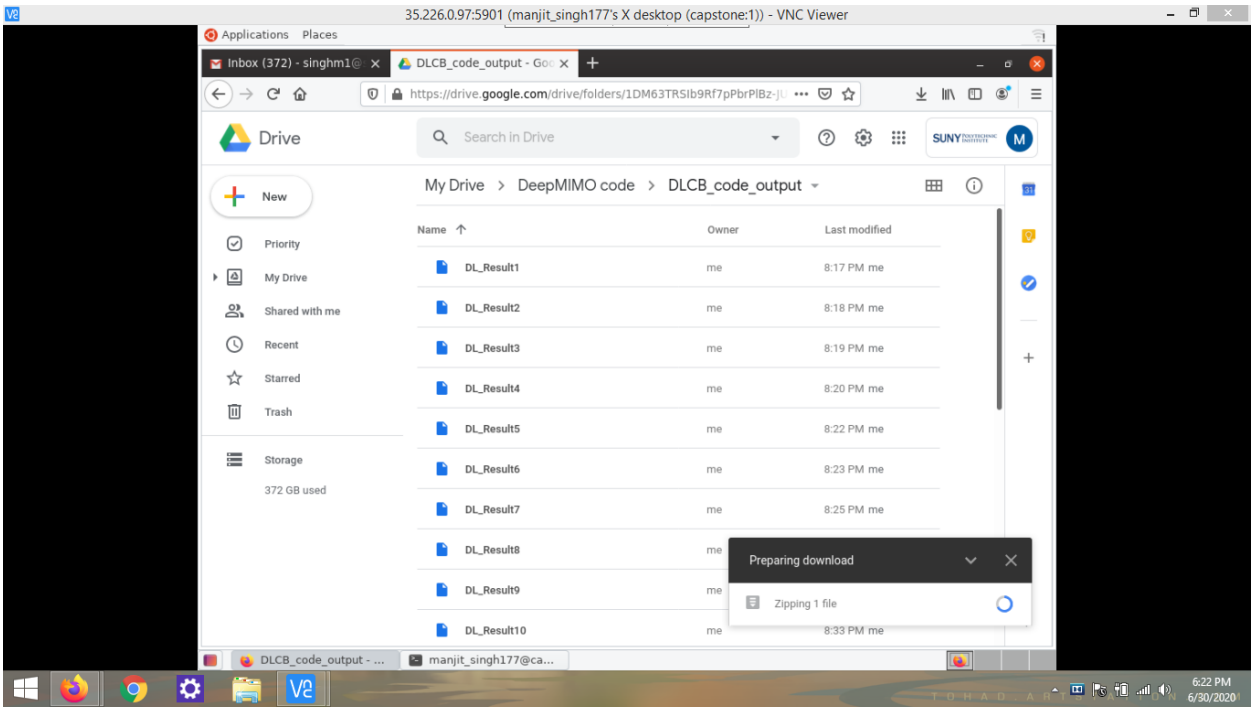
Run the dataset generation code and wait for it to complete. The file will be saved as dataset.mat



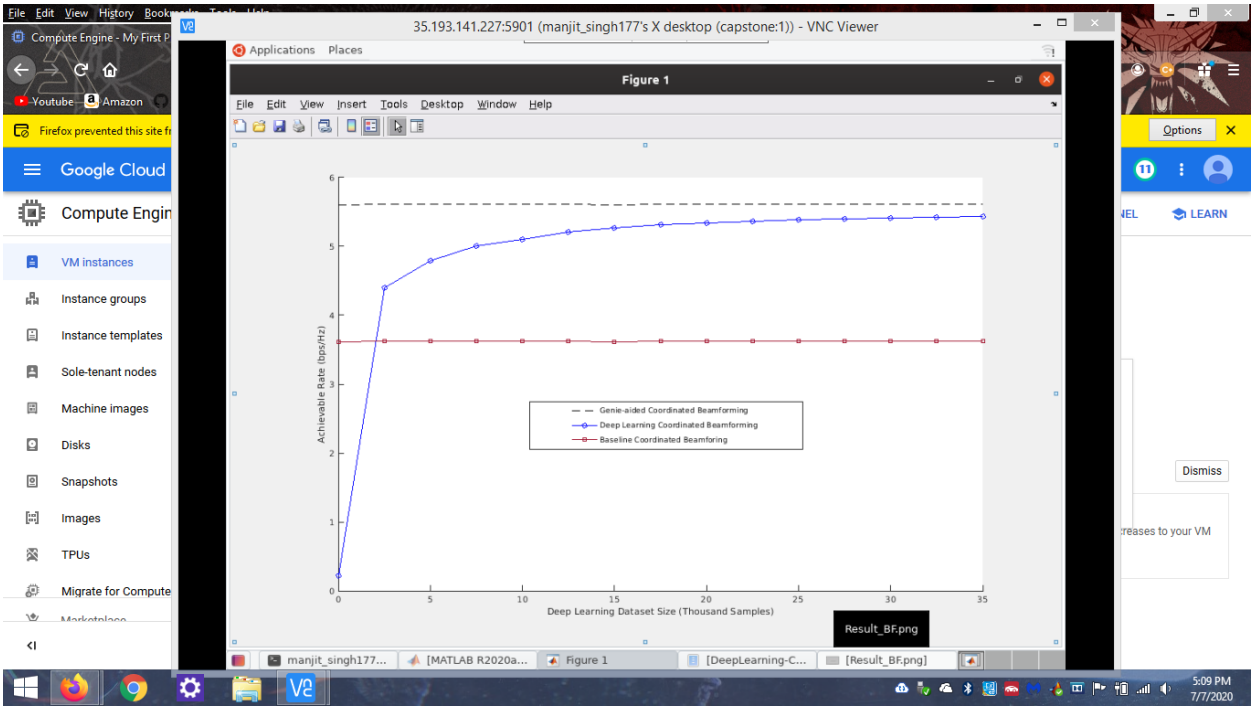
Now run the DL beamforming code to generate the Deep learning model. You will end up with an input and output.mat file, which can be seen in the MATLAB workspace on the bottom left.



Run the DL_model_python_code.py inside TensorFlow to begin building and testing your model. This will be 100 rounds and 10 epochs. This should take about 30min-45minutes to train.



Once the model has been trained these files will appear in the DLCB_code_output folder. You will need to import these as ".mat" format before you can run the figure_generator.m code to plot your output.



Result of the graph should appear like this. The x-axis is the size of the dataset, which as the dataset size increases the model becomes more accurate and is able to reach the highest achieving rate (y-axis) at the upper bound /genie-aided beamforming solution which knows the most optimal

beamforming vectors (A. Alkhateeb, 2018). Compared to the baseline beamforming algorithm, which uses the combining of the uplink pilot signature and the codebook/omni beams to search for all the possible beamforming vectors, which is an exhaustive approach and can take up large amount computation time, this makes it inefficient in supporting wireless applications that have high throughput and mobility requirements (A. Alkhateeb, 2018). Although the baseline solution in a few cases can achieve the highest achievable rate in special occasions it is considered inefficient compared to the proposed baseline solution (A. Alkhateeb, 2018).

Evaluation.

This section will use the results of the graph from this project and compare it to other classification approaches that use the same dataset.

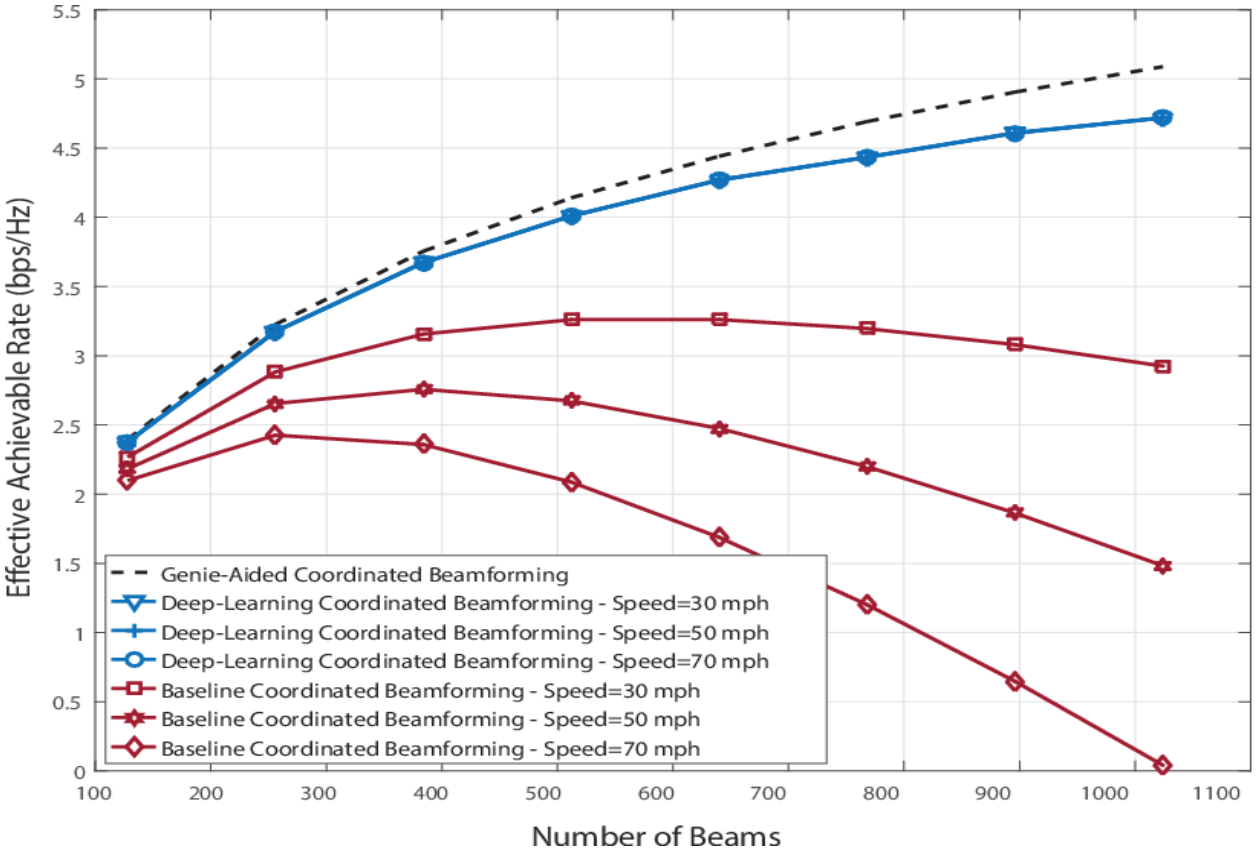


Image source: https://www.semanticscholar.org/paper/Deep-Learning-Coordinated-Beamforming-for-Wave-Alkhateeb-Alex/cc779ba71eabdf68944051eea852a24eaf79092c/figure/8

The velocity of our coordinated beamforming solution compared to this graph was 50 mph and seems to approach the genie-aided solution along with the other DL beamforming solutions of velocity 30, and 70 mph. The achievable rate(y-axis) increases as the number of beams(x-axis) increases compared to the baseline solution, which begins to fall at all velocities as the number of beams increase (A. Alkhateeb, 2018). This is the benefit of deep learning as the more data you give it the better the accuracy and performance as compared to searching for all possible combinations through codebook beam and omni beam combinations, which takes away from the

efficiency due to latency and computation time, this affects the rate significantly (A. Alkhateeb, 2018).

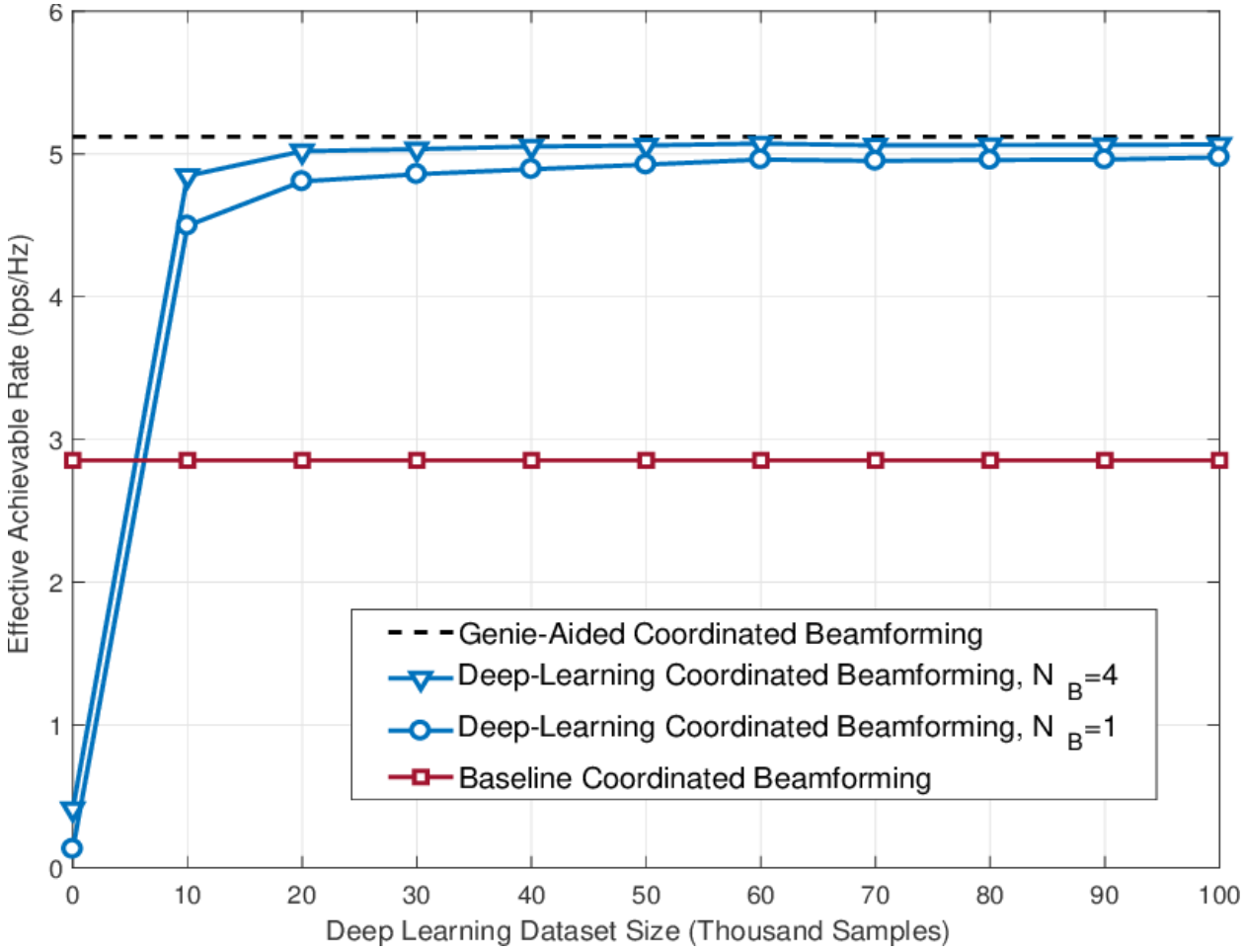


Image source: https://www.semanticscholar.org/paper/Deep-Learning-Coordinated-Beamforming-for-Wave-Alkhateeb-Alex/cc779ba71eabdf68944051eea852a24eaf79092c

This graph shows a LOS and NLOS (line of sight and non-line of sight) scenario with 4 base stations on a vehicle moving at 30 mph. In our scenario 3 base stations were used at 50 mph and would fall right in the middle between the two beamforming solutions and would approach the effective achievable rate of the upper bound Genie-aided solution (A. Alkhateeb, 2018). On the graph created from this project the scenario used was the "01" scenario, however the beamforming solution is effective in both LOS and Non-LOS scenarios regardless and the velocity is 20 miles faster in the "01" scenario than the one used in the "LOS/NLOS" scenario graphed above. The use of uplink pilot signature and omni-quasi/ omni beams to train and then switch back and forth from learning and predicting is what allows the proposed coordinated beamforming solution to benefit in both LOS/NLOS scenarios since it allows the NN to adapt, whereas if GPS data were used this would not be the case (A. Alkhateeb, 2018).

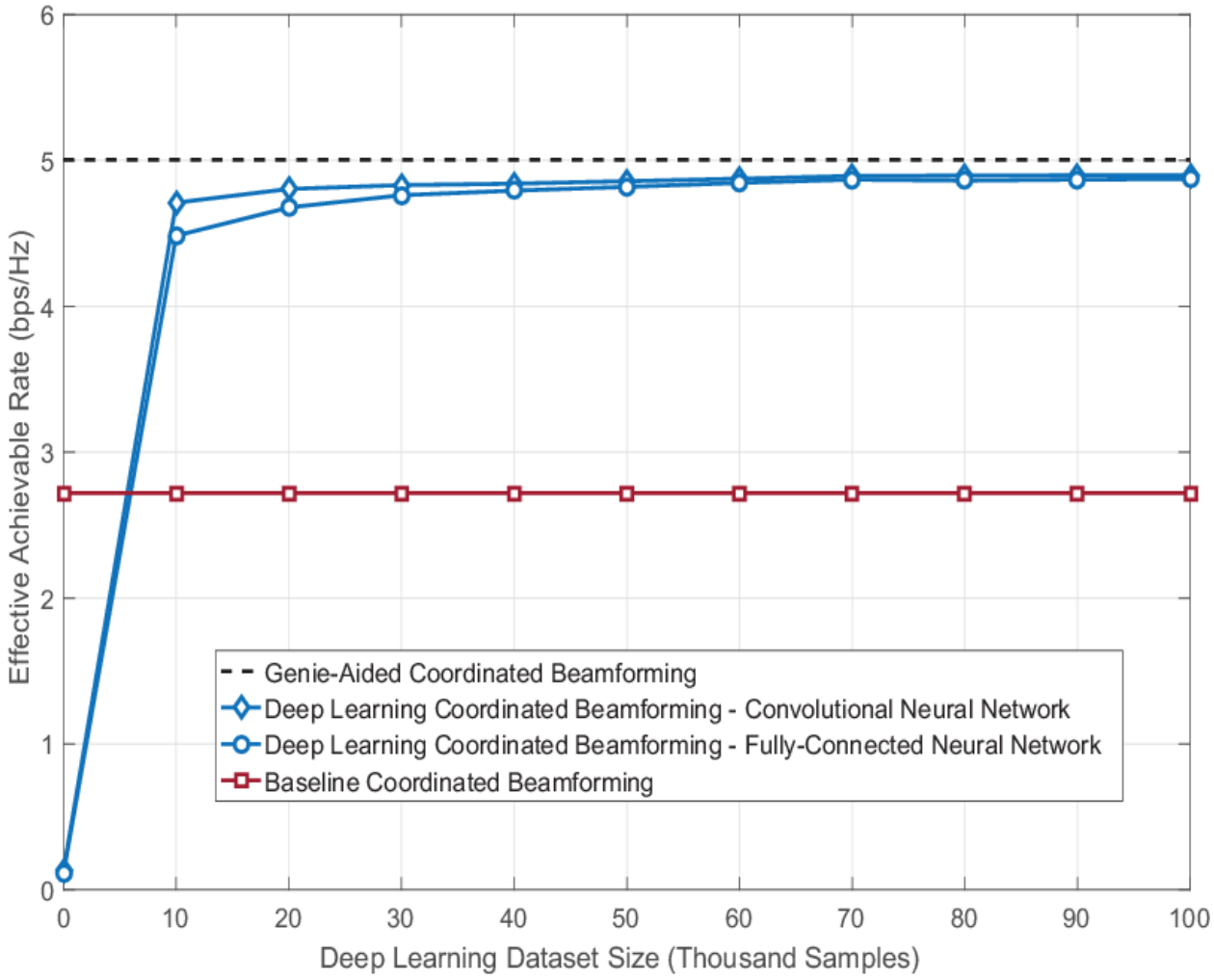


Image source: https://www.semanticscholar.org/paper/Deep-Learning-Coordinated-Beamforming-for-Wave-Alkhateeb-Alex/cc779ba71eabdf68944051eea852a24eaf79092c

This graph shows a Convolutional NN compared to a fully connected NN. The results are very similar as they both approach the upper bound as the size of the dataset increases. A fully connected NN means that each layer is connected, as in a hidden layer with an output becomes an input in the next layer (A. Alkhateeb, 2018). Our graph indicates that it would be right alongside these two beamforming solutions, similar in result. The two solutions on the graph belonging to the author above and the reproduced graph from this project indicate that they are very similar in efficiency in reaching the highest achievable rate compared to the baseline solution, however the CNN is slightly more efficient as it achieves the same results if not slightly better with less the complexity and less parameters, 754k vs 1048k of the fully connected network (A. Alkhateeb, 2018). This could be according to the authors due to how the CNN extracts local information through small filters (A. Alkhateeb, 2018). These filters capture, the correlation between adjacent samples in the (Orthogonal Frequency Division Multiplexing) OFDM sequence, which can help extract information with much lower complexity than simply the technique used by the fully connected NN (A. Alkhateeb, 2018). So the beamforming solution is efficient, however the CNN is just as

efficient if not better with less complexity meaning that other architectures could bring similar if not better results than even the coordinated beamforming solution (A. Alkhateeb, 2018).

Conclusion and Future Work.

In conclusion the purpose of this report is to analyze two papers and reproduce the steps and compare them to the authors by creating a generic dataset that is defined from a ray tracing scenario and adjustable channel parameters (Alkhateeb, {2019}). Then to take this dataset and generated Deep learning model to be trained inside TensorFlow using Google Collaboratory. Once the output files were received the results were imported back into the cloud VM and converted to ".mat" format so that it can be read by the figure generator code. This code would plot the results of our beamforming solution so that we may compare it to another solution that used the same dataset. Overall, this solution reaches the highest achieving rate with little computation time compared to the baseline coordinated solution (A. Alkhateeb, 2018). This was due to the coordinated beamforming solution using only the omni beam pattern to find the best beamforming vectors as compared to the baseline solutions exhaustive approach of searching for every combination of beamforming vector through the combining of the uplink pilot training and codebook beams (A. Alkhateeb, 2018). The drawback of this is that beam training takes computation time and makes it less efficient in highly mobile applications (A. Alkhateeb, 2018). The coordinated beamforming solution results were slightly improved by the CNN solution (A. Alkhateeb, 2018). This was due to the local filters on the CNNs that are able to make correlations between adjacent samples of the OFDM sequence, this can extract valuable information with much lower complexity (A. Alkhateeb, 2018). Overall, the coordinated beamforming solution makes many improvements in efficiency in its approach compared to the baseline solution, however the same can be said about the CNN model, which brings the same results as the Coordinated beamforming solution, but with less complexity, which motivate further research in other Neural network model architectures for the mmWave and massive MIMO systems (A. Alkhateeb, 2018).

We plan to study the 5G systems security and adopt our current research work [12-23] to this new research domain particularly the underlying security issues of the mmWave and massive MIMO systems.

References questo paper non esiste

[1] A. Alkhateeb, S. A. (2018). Millimeter wave beam prediction based on multipath signature. 42.

[2] Alkhateeb, A. ({2019}). {{DeepMIMO}: A Generic Deep Learning Dataset for Millimeter Wave and Massive {MIMO} Applications}. {Proc. of Information Theory and Applications Workshop (ITA)}, 1-8.

[3] Alkhateeb, A. (2019). semanticscholar. Retrieved from semantic scholar: https://www.semanticscholar.org/paper/DeepMIMO%3A-A-Generic-Deep-Learning-Dataset-for-Wave-Alkhateeb/d1d8980d04d411c314910d1926f3bbaac46c2197#extracted

[4] Donges, N. (2017, november 15). Why is Deep Learning taking off? Retrieved from machinelearning: https://machinelearning-blog.com/2017/11/03/erster-blogbeitrag/

[5] Gupta, R. (2020, July 11). Getting started with Neural Network for regression and TensorFlow. Retrieved from Medium : https://medium.com/@rajatgupta310198/getting-started-with-neural-network-for-regression-and-tensorflow-58ad3bd75223

[6] InSite, T. R. (n.d.). Wireless-insite. Retrieved from Remcom Wireless: https://www.remcom.com/wireless-insite

[7] Knocklein, O. (2019, June 15). Classification Using Neural Networks. Retrieved from Towards DataScience: https://towardsdatascience.com/classification-using-neural-networks-b8e98f3a904f

[8] Malik, F. (2018, October 09). Supervised Machine Learning: Regression Vs Classification . Retrieved from medium.com: https://medium.com/fintechexplained/supervised-machine-learning-regression-vs-classification-18b2f97708de

[9] Pokharna, H. (2016, July 28). The best explanation of Convolutional Neural networks on the internet. Retrieved from Medium: https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8

[10] Ramsundar, B., & Zadeh, R. (. (n.d.). Tensorflow for Deep Learning. Retrieved from oreilly Web site: https:www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04.html

[11] Yegulalp, S. (2018, june 18). What is TensorFlow? The machine learning library explained. Retrieved from Scribbr: https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html

[12] Kholidy, H.A., Fabrizio Baiardi, Salim Hariri: 'DDSGA: A Data-Driven Semi-Global Alignment Approach for Detecting Masquerade Attacks'. The IEEE Transaction on Dependable and Secure Computing, 10.1109/TDSC.2014.2327966, pp:164–178, June 2015.

[13] Kholidy, H.A., Fabrizio Baiardi, "CIDD: A Cloud Intrusion Detection Dataset For Cloud Computing and Masquerade Attacks ", in the 9th Int. Conf. on Information Technology: New Generations ITNG 2012, April 16-18, Las Vegas, Nevada, USA. http://www.di.unipi.it/~hkholidy/projects/cidd/

[14] Kholidy, H.A., Fabrizio Baiardi, "CIDS: A framework for Intrusion Detection in Cloud Systems", in the 9th Int. Conf. on Information Technology: New Generations ITNG 2012, April 16-18, Las Vegas, Nevada, USA. http://www.di.unipi.it/~hkholidy/projects/cids/

[15] Kholidy, H.A., Baiardi, F., Hariri, S., et al.: 'A hierarchical cloud intrusion detection system: design and evaluation', Int. J. Cloud Comput., Serv. Archit. (IJCCSA), 2012, 2, pp. 1–24.

[16] Kholidy, H.A., "PH.D. Thesis: Cloud Computing Security, An Intrusion Detection System for Cloud Computing Systems". https://pdfs.semanticscholar.org/cf8a/14dc638480dbc5304824dd99a631d917d3fe.pdf

[17] Kholidy, H.A., “Autonomous mitigation of cyber risks in the Cyber–Physical Systems”, Future Generation Computer Systems, Volume 115, 2021, Pages 171-187, ISSN 0167-739X, https://doi.org/10.1016/j.future.2020.09.002. http://www.sciencedirect.com/science/article/pii/S0167739X19320680

[18] Kholidy, H.A., Abdelkarim Erradi, Sherif Abdelwahed, Fabrizio Baiardi, "A risk mitigation approach for autonomous cloud intrusion response system", in Journal of Computing, Springer, DOI: 10.1007/s00607-016-0495-8, June 2016.

[19] Kholidy, H.A., Abdelkarim Erradi, “A Cost-Aware Model for Risk Mitigation in Cloud Computing SystemsSuccessful accepted in 12th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA), Marrakech, Morocco, November, 2015.

[20] Kholidy, H.A., Ali Tekeoglu, Stefano Lannucci, Shamik Sengupta, Qian Chen, Sherif Abdelwahed, John Hamilton, “Attacks Detection in SCADA Systems Using an Improved Non-Nested Generalized Exemplars Algorithm", the 12th IEEE International Conference on Computer Engineering and Systems (ICCES 2017), December 19-20, 2017.

[21] Qian Chen, Kholidy, H.A., Sherif Abdelwahed, John Hamilton, "Towards Realizing a Distributed Event and Intrusion Detection System", the International Conference on Future Network Systems and Security (FNSS 2017), Gainesville, Florida, USA, 31 August 2017. Conference publisher: Springer. “Industrial control system (ics) cyberattack datasets, http://www.ece.uah.edu/~thm0009/icsdatasets/ PowerSystem_Dataset_README.pdf

[22] Kholidy, H.A.,, Abdelkarim Erradi, Sherif Abdelwahed, Abdulrahman Azab, “A Finite State Hidden Markov Model for Predicting Multistage Attacks in Cloud Systems", in the 12th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC), Dalian, China, August 2014.

[23] Kholidy, H.A., “Detecting impersonation attacks in cloud computing environments using a centric user profiling approach”, Future Generation Computer Systems, Volume 115, issue 17, Decmenr 13, 2020, Pages 171-187, ISSN 0167-739X, https://doi.org/10.1016/j.future.2020.12
https://www.sciencedirect.com/science/article/abs/pii/S0167739X20330715