Channel Assignment for Multi-User Cognitive Radio using Deep Learning and Linear Programming

A Project Report

submitted by

NIKHIL V

in partial fulfilment of the requirements

for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

June 2021

DEPARTMENT OF ELECTRICAL ENGINEERING

Indian Institute of Technology, Madras

India - 600036

CERTIFICATE

This is to certify that the thesis titled **Channel Assignment for Multi-User Cognitive Radio using Deep Learning and Linear Programming**, submitted by **NIKHIL V**(Roll Number: *EE19M022*), to the Indian Institute of Technology, Madras, for the award of the degree of Master of Technology, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Sheetal Kalyani

Project Guide,
Associate Professor,
Dept. of Electrical Engineering,
IIT Madras, 600 036.

Place: Chennai

Date: June 18, 2021

ACKNOWLEDGEMENTS

I am greatly indebted to Dr. Sheetal Kalyani for guiding me through the entire course of my M.Tech. Project. She always took the time and effort to discuss the problem and to suggest different methods to experiment. Her valuable remarks always gave new directions to my project.

I am thankful to all the professors whose courses helped me improve my knowledge in Wireless Communication and Signal Processing through the two years of my M.Tech. Program. Their classes always inspired me to think beyond classrooms into more practical scenarios.

A special thanks to Thulasi, who was my fellow bandit in doing this project. This project would not have been possible without her contributions and insightful observations. A special word of thanks to all the wonderful people I met at IIT without whom life would not have been the same.

ABSTRACT

Keywords: Cognitive Radio, Optimal Channel selection, Hungarian Algorithm, Linear Programming, Deep learning, Fairness

Cognitive Radio(CR) technology is widely being used in recent years to improve the spectral efficiency of Wireless Communication systems by opportunistically accessing a communication channel when its licensed user is not transmitting. The CR system's performance can be enhanced if the secondary users cooperatively share information regarding the primary channel traffic. A centralized approach for assigning channels to users using the information obtained from secondary users is proposed in this thesis. The resource allocation of channels to users can be modelled as a linear sum assignment problem(LSAP). An algorithm based on deep learning implementation of LSAP is developed to determine the most suitable match between channels and users when multiple spectral holes are available to multiple secondary users. Convolution neural networks were used to analyze the information from SUs and make decisions regarding channel assignment. Fairness in the assignment is also considered an essential element in this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid advancements in information and communication technology have increased the number of devices using the licensed spectrum of radio frequency. In past years, the strategy used for spectrum allocation was fixed spectrum assignment policy in which a part of the spectrum was designated for a particular application. Due to the dramatic increase in the number of users and the spectrum being a scarce resource, fixed assignment policies became less effective in meeting current demands. In 2002, the Federal Communications Commission(FCC) Spectrum Policy Task Force published a report [1] in which the under-utilization of the licensed spectrum was evident; a large portion of the assigned spectrum is used sporadically. Dynamic spectrum allocation was a feasible alternative to effectively utilize the infrequently active licensed spectrum of the primary user and increase the efficiency of spectrum usage.

Cognitive Radio(CR) is the crucial technique that enables dynamic spectrum access, thereby providing a means to share the licensed spectrum with the secondary users(SU) when the primary user(PU) is not active [2]. Cognitive Radio is an intelligent device capable of understanding the surrounding environment and varies its internal parameters by analyzing variation in the incoming radio frequency stimuli. It offers several functionalities like spectrum sensing, spectrum decision, spectrum sharing and spectrum mobility[3, 4]. Determining the inactive portion of the spectrum is called spectrum sensing [5]. After the opportunity in the spectrum is detected, the

CR can make the decision on allotting the observed vacant channel to any of the requesting users based on internal policies [6]. In spectrum sharing, efficient allocation of the channel to the secondary user while causing minimum interference to the primary user [7]. Two transmission models which were used to allow SU to do dynamic spectrum access in *underlay* [8] and *overlay* [9] models. In the overlay model, the SUs are allowed to communicate alongside the primary user as long as the interference caused does not degrade the communication equality of the primary user. In the underlay model, the SU can exclusively access the channel when the primary user is idle. Once a licensed user is detected, the secondary user must vacate the channel, and the channel must be made available for the primary user. This functionality is called the spectrum mobility in cognitive radio [10] .

Different spectrum sharing models are developed to fulfil various capabilities like open sharing, hierarchical access and dynamic exclusive usage models[11]. Another necessary functionality is spectrum management which evenly satisfies the requirements of both primary and secondary users. The typical spectrum sensing approach for a single user system is energy detection, cyclostationary detection, matched filtering and wavelet detection. As mentioned in [12], there is always a trade-off between the optimum time for spectrum sensing and the throughput of the system. The more time involved in spectrum sensing, the lesser the time left for actual transmission and directly impacted the maximum throughput. Thus, efficient techniques have to adopted to provide minimum interference without compromising much of the transmit duration. With a reduction in the number of sense operations, more time can be used for transmission.

The cooperation among users is shown to have enhanced the performance in cognitive radio systems [13, 14, 15]. Whenever cognitive users detect spectral holes, the status of the channel is shared so that excessive interference and collision risk with the primary user is reduced [16, 17]. A centralized algorithm to effectively combine the sensing decisions of various nodes is proposed in [18]. A scheme in which a node listens to their neighbouring node and passes the information on the collision is proposed in [19]. A centralized framework is proposed in [20], in which a central unit is responsible for coordination between the secondary users in order to avoid inter SU collision and also obtains feedback data from the SU regarding the channel traffic behaviour.

In this thesis, we build on top of the CR architecture proposed in [20] and develop an assignment algorithm that makes use of the information already present with the central node. The assignment technique ensures that when multiple users are waiting for opportunistic access to multiple spectrum holes, the best matching is done between the users and the channels. Furthermore, when the number of users is more than the available spectrum holes, fairness in allocation is also considered while doing the assignment.

The remainder of this thesis is organized as follows:

Chapter 2 introduces the system model that is used for this work. A brief outline of the Cognitive Radio technology is presented along with the centralised approach used to allocated spectrum holes to unlicensed users. The nature of the input, output and the relation between them is demonstrated here.

Chapter 3 briefly explains the Assignment problem and discusses the reason why it is a relevant problem. An overview of the Deep Learning technique and its applicability in this particular setting is also included in this chapter.

In *Chapter 4*, the proposed optimal channel assignment strategy is explained using different channel conditions. An algorithm was put forward to estimate the assignment decision when the channel cost is not initially available. A novel strategy is introduced to avoid multiple collisions while assigning channels.

The simulation setup and the obtained results are explained in *Chapter 5*. The accuracy of the proposed method of assignment is compared with other methods for different network dimensions.

Chapter 6 is the summary of the work done and provided some concluding remarks and thoughts on further enhancements possible.

Chapter 2

System Model

2.1 Introduction

The spectrum occupancy of various channels in a wireless communication network can be understood by observing an indicative time versus frequency diagram as in Figure. 2.1. From the figure, it could be observed that the channels are not having a consistent traffic behaviour just like the case in practical wireless scenarios. Whenever a primary user is not occupying its designated channel, the channel remains idle until it starts transmitting again. This is known as a *spectrum hole*.

Cognitive Radio networks uses this under-utilization of licensed channels by the primary users in advantage of unlicensed secondary users to improve the overall spectral efficiency of the wireless communication network. In the Figure. 2.1 , the red line illustrates how a secondary user utilizes the spectrum holes by hopping from one channel to another whenever the primary user starts using its respective channel.

In real scenarios, multiple SUs will be competing to attain access of the available spectrum holes. Therefore an efficient strategy need to be devised to avoid competition between multiple secondary users. In [20], a centralized approach was introduced to monitor the traffic in the given spectrum and assign available channels to users so that the collision among multiple SUs and between PUs and SUs are minimized.

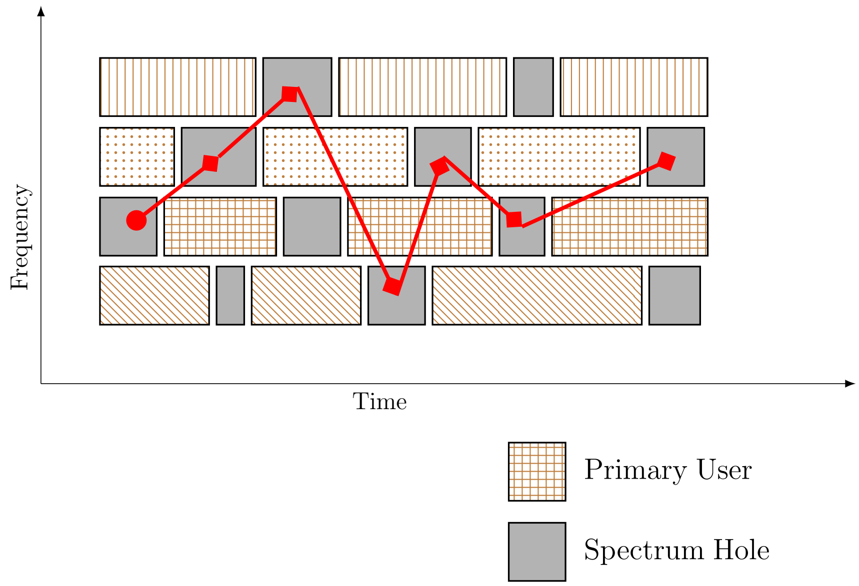

Figure 2.1: Spectrum utilization by primary user

2.2 Centralized Model

In the assumed Cognitive Radio Network, a set $\mathcal{N}$ of primary users and a set $\mathcal{M}$ of secondary users is considered such that $|\mathcal{N}| =$ N and $|\mathcal{M}| =$ M. All the users in the set $\mathcal{N}$ have designated channels which they could occupy when communicating. There are N channels available for opportunistic usage by SUs. The set $\mathcal{N}$ contains both active PUs ($\mathcal{N}_a$) which are using the designated channels and idle PUs ($\mathcal{N}_i$) which are not currently using its licenced channel. These sets are mutually exclusive and collectively exhaustive i.e., $\mathcal{N}_a \cup \mathcal{N}_i = \mathcal{N}$ and $\mathcal{N}_a \cap \mathcal{N}_i = \emptyset$. When an idle channel is up for communication, it moves from the set $\mathcal{N}_i$ to the set $\mathcal{N}_a$.

The centralized framework ensures the efficient working of the system by using a central unit/node to which all the SUs report about the primary channel traffic information. The central node coordinates the data transmission process of secondary devices. The secondary users set $\mathcal{M}$ can also be divided into four disjoint sets de-

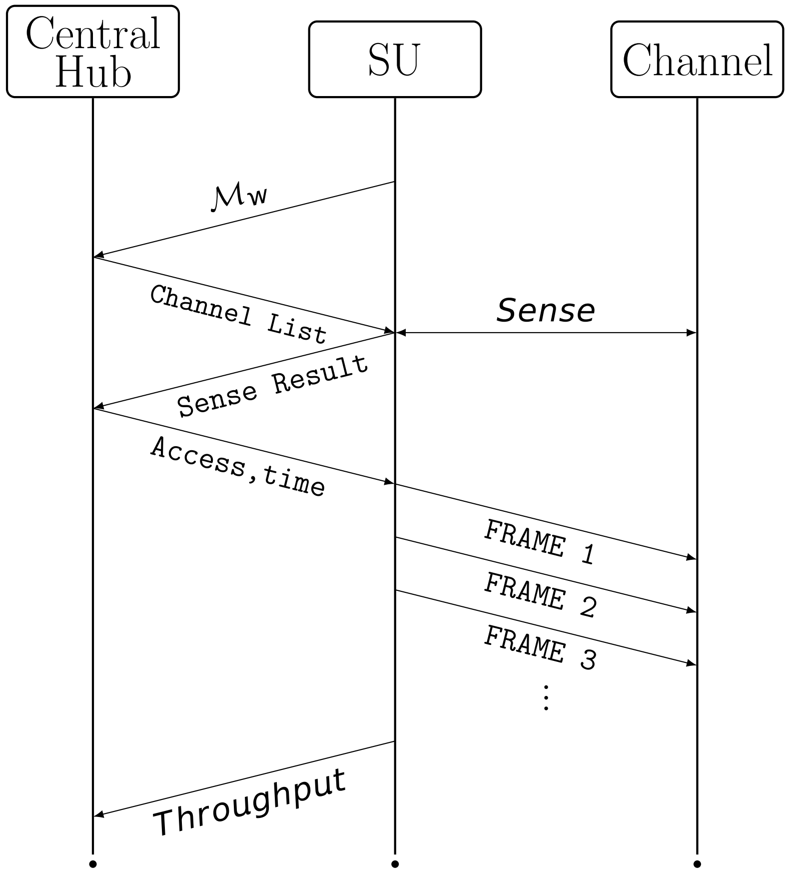

Figure 2.2: CRN connection Time Line representation

pending on the state of each SU. The idle users belong to $\mathcal{M}_i$, the devices which are currently sensing the primary channel are in $\mathcal{M}_s$, the users which are waiting to be serviced in $\mathcal{M}_w$, and the users who are actively transmitting form the set $\mathcal{M}_a$. The interactions between the central hub and secondary users can be summarised using the Fig. 2.2.

2.3 Channel Assignment Model

The central hub supervises the channel assignment to SUs such that the throughput efficiency of the system is optimal. Each device reports to the central hub the information regarding its cost to use a given channel. After attaining the cost of using every available primary channel by each user in the $\mathcal{M}_w$ set, the central node should assign the channels to users based on some optimality condition. In this work, the cost is defined in terms of the Packet Reception Error at the receiver. Hence the optimal assignment should be such that the total cost of the channel allocation is minimised.

2.3.1 Cost and Decision Matrix

Cost Matrix: The user to channel cost information observed by the central node is written in the form of a matrix. This matrix is the Cost Matrix. The packet reception error is represented using a matrix with a row dimension equal to the number of available channels for opportunistic access and a column dimension equal to the number of users in the $\mathcal{M}_w$ set.

Decision Matrix: The decision presented by the central unit is written in the form of a logical matrix i.e., all the entries are either 0 or 1. An entry is 1 when the channel represented by the row is assigned to the user represented by the column.

$$\begin{aligned} \sum_{i=1}^{N} x_{ij} \leq 1; \qquad & i = 1, 2, \ldots, N \\ \sum_{j=1}^{N} x_{ij} \leq 1; \qquad & j = 1, 2, \ldots, N \\ x_{ij} \in \{0, 1\}; \qquad & i, j = 1, 2, \ldots, N \end{aligned} \tag{2.1}$$

The constraints specified in Eqn. 2.1 ensures that no user is given access to more than one channel at a time, and every channel is assigned to utmost one channel.

Chapter 3

Linear Programming and Deep Learning Overview

3.1 Linear Programming

Linear programming(LP) or Linear optimization is the method used to allocate scarce resources subject to some linear optimization function. All the relations among variables related to resources is also linear. Optimality in LP is often defined as maximizing profit or achieving the lowest cost for an assignment.

3.1.1 Linear Sum Assignment Problem

Linear Sum Assignment Problem (LSAP) is popular linear programming and combinatorial optimization problem where assigning jobs to people is done most efficiently. In simple terms, for a given matrix C with dimension $n \times n$, the optimization problem would be to match each row to a different column so that the sum of all the selected entries is minimum. The assignment is to select n elements from the matrix such that each row and each column has exactly one element, and the sum of these n elements is minimum.

3.1.1.1 Numerical Example

Consider the cost matrix in Equation 3.1. Observing the entries of the cost matrix, it can be seen that for user 1 (column 1), channel 5 has a value less than channel 1, i.e.,

$C_{51} < C_{11}$. However, the assignment strategy that minimizes the overall sum cost assigns channel 1 to user 1 rather than assigning it to channel 5. Hence, a straight forward linear approach cannot be employed here.

$$
\begin{aligned}
Cost = &\begin{bmatrix} 3 & 31 & 45 & 27 & 33 \\ 22 & 84 & 38 & 39 & 60 \\ 91 & 85 & 37 & 30 & 31 \\ 55 & 16 & 74 & 57 & 85 \\ 2 & 53 & 19 & 26 & 34 \end{bmatrix} \\
&\Downarrow \\
Decision = &\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{3.1}
$$

3.1.1.2 Weighted bipartite matching problem

The assignment problem can alternatively be expressed using the Graph theory model. Let G be a graph represented by G= (U, V; E) where U is the vertex representing each row, and V is the vertex representing each column. The elements C_{ij} of the cost matrix C is represented as an *edge* from the ith vertex in U to the jth vertex in V. Now, the optimization that is to be done for this *weighted bipartite graph* to find a subset of the edges E such that each vertex belongs to exactly one edge and the sum of the costs provided by these edges is the minimum.

In Figure 3.1, the red edges represent the optimal assignments that minimize the total cost. It can be observed that each vertex has only a single red edge associated with it.

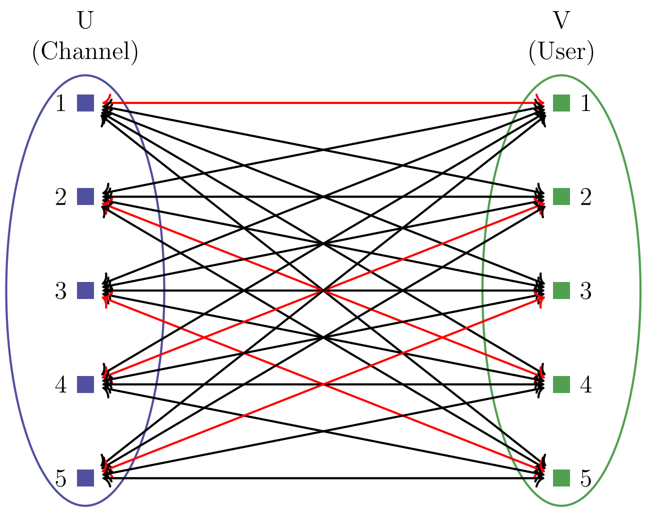

Figure 3.1: Bipartite representation of Channel-user cost relation

3.1.1.3 Hungarian Algorithm

A straightforward approach would be to do a brute force search over all possible assignment combinations and select the assignment which gives the least total cost. Generating all combinations is done by forming all possible permutations of the indices so that no row and column are used more than once. While this approach seems to work fine for matrices with low order, it does not scale. For a matrix of dimension $n \times n$, the total number of permutation operations needed to obtain the optimal solution is $n!$ i.e., this method operates in $O(n!)$ time. For an 8×8 matrix, the total number of permutation operations need would be *3628800*. If the time required is optimistically taken as 1 millisecond per operation, then it will take an hour to traverse over all combinations. A 16×16 matrix would take more than *663 years* to generate the decision.

In 1955, Harold Kuhn put forward an algorithm called *the Hungarian method for Assignment problem* that could solve the assignment problem faster [21]. The computational advantage in this technique is gained by considering the problem in connection with duality in Linear Programming [22]. The work was later reviewed by James Munkres and observed that the time complexity of the method was strongly polynomial [23]. While the original approach solved the problem in $O(n^4)$ time, two works [24],[25] independently modified the algorithm to work in $O(n^3)$ time, which is a significant improvement over the $O(n!)$ for higher-order matrices.

3.2 Deep Learning based Approach

In a real-time application as the Cognitive Radio, the computational time of $O(n^3)$ is still not acceptable as the sensed spectrum hole might not remain vacant for so long. The CR system should be able to make the channel assignment decisions real quick. So, the conventional implementation of the Hungarian Algorithm cannot be used in this problem setting. The deep learning approach seemed to be an efficient candidate that could replace the existing implementation of the Hungarian algorithm [26].

In the work [27], a deep learning implementation of the Hungarian Algorithm is proposed. Since the Hungarian Algorithm can neither be posed as a regression problem nor a classification problem, the process of deciding the assignment matrix is decomposed into multiple sub-assignment problems Figure.3.2. Hence several neural Networks are used to get each column of the Decision matrix. Since this approach involves multiple Neural Networks predicting columns of the Decision matrix, there are chances of encountering collisions, i.e., More than one user will be assigned the same channel. In such scenarios, a low complexity greedy collision avoidance strategy is employed to make a decision. The implementation shows that the implementation could replicate the results of the Hungarian Algorithm at a faster rate, with a slight drop in accuracy.

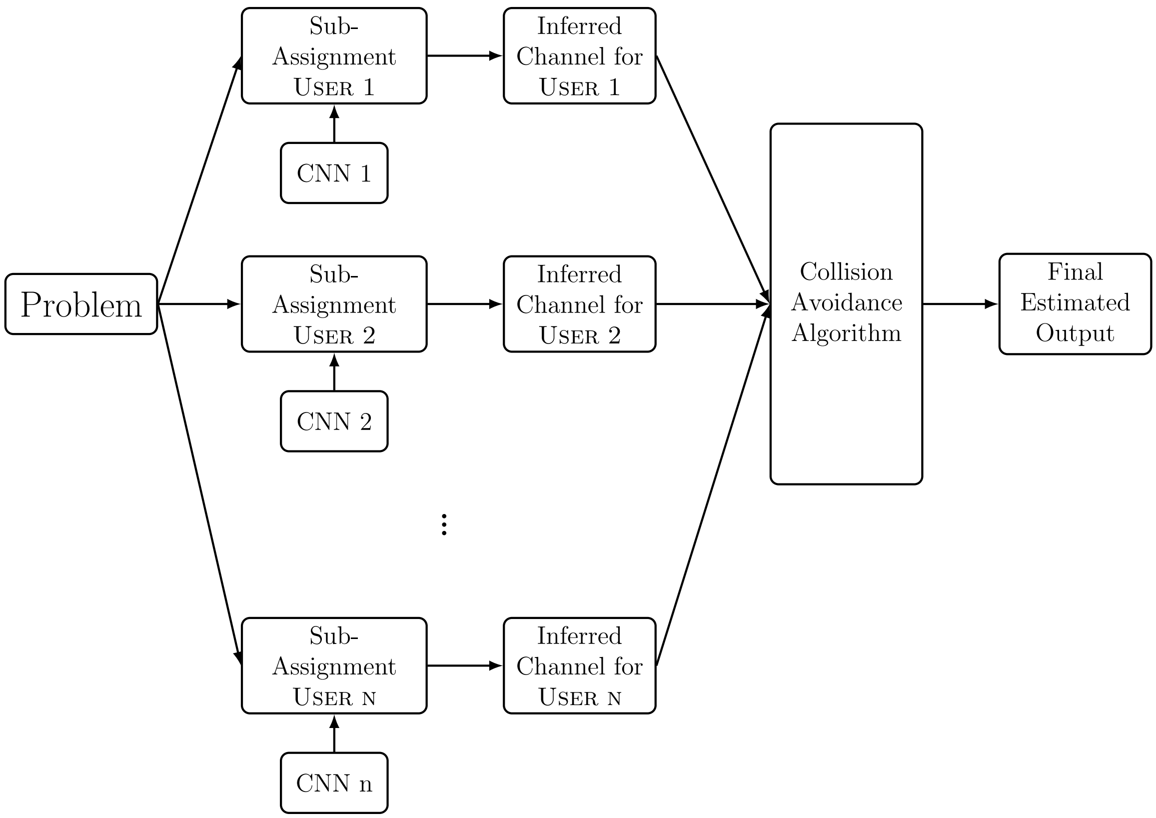

Figure 3.2: Decomposition of Assignment Problem into multiple sub-assignment problems

Chapter 4

Proposed Approach

4.1 Estimation of Cost and Channel Assignment Using Pre-Trained Network

The problem with channel allocation using the technique mentioned here is that, in practical scenarios, the complete cost information will not be available to the central node at all instants. So, some of the assignments should be made using partial measurements of the known costs. Initially, the cost matrix would be initialized using zero initialization; then, this matrix is fed into the pre-trained neural network to get the initial prediction of the decision matrix $\tilde{D}_{ij}$. When an all-zero cost matrix is given as input to the network, it will output a random assignment for the decision matrix and it will depend only on the Network's weights. The central node will assign the channels to users according to the initial decision for T instants. After transmitting for T instants, the central node will be having a noisy estimate of the cost values of these entries $\tilde{C}_{ij}$. The i, j values are inferred from the preceding decision matrix. This estimate is used to update the initial cost matrix. Now, when the updated cost matrix is given as the input to the network, the new decision matrix will have random assignment again. Now, the i, j values would not be the same as the ones obtained from the previous decision estimate $\tilde{D}_{ij}$. This is because the new cost matrix has zeros at all entries except for the updated positions. The positions with zero entries are likely to be assigned since the objective of the network is to minimize the total cost. So, after every iteration, the channel will be given to different users, thereby obtaining the complete cost information in a few iterations.

Algorithm 1 Cost Matrix updating using Neural Networks

Training Dataset Generation: Generate random C matrix with dimension (N, M) and the corresponding Decision X using Hungarian Algorithm ▷ $|\mathcal{N}_i| = N$ ▷ $|\mathcal{M}_w| = M$

for $i = 1, 2, \cdots M$ **do**
    $f^{(L)}(f^{(L-1)}(\ldots f^{(0)}(C))) = X_{ij}$ ▷ Training Networks for user i

Deploying trained Network when cost is unknown
Initialize Cost $\tilde{C} = 0 \quad \forall i, j$
while (Secondary Users Transmitting) **do**
    $f^{(L)}(f^{(L-1)}(\ldots f^{(0)}(\tilde{C}))) = X$ ▷ Optimal Decision Estimation
    Transmit using the decision X for T instants
    for $t = 1, 2, \cdots T$ **do**
        Obtain packet reception error
    Calculate cost using reported packet reception error
    Update cost using the Estimates
    $mask \leftarrow ones(N, M) - X$
    $\tilde{C} \leftarrow (\tilde{C} * mask) + (NoisyCost * X)$

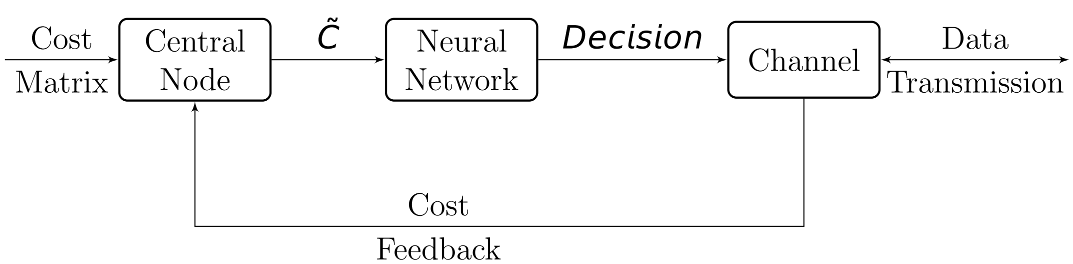

Figure 4.1: Cost Estimation using Pre-trained Network

4.2 Numerical Example

Let the true cost matrix of the underlying System be

$$\textbf{True Cost} = \begin{bmatrix} 4.6 & 7.25 & 3.17 & 3.8 \\ 5.43 & 8.5 & 2.89 & 1.4 \\ 3.06 & 9.1 & 6.06 & 5.18 \\ 6.61 & 5.33 & 7.91 & 3.1 \end{bmatrix} \tag{4.1}$$

The central node does not have the knowledge of the true cost matrix at the beginning of transmission. So initially, the cost Matrix will be zero-initialized.

$$\textbf{Initial Cost} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.2}$$

When this initial cost matrix is presented as the input to the pre-trained network, it will decide depending on the network's weights alone. This decision is not optimal and can't be used to allocate the channel to a user for the data transmission. The obtained decision matrix will help get the underlying true cost Matrix. For a given pre-trained Network, let the initial assignments be

$$\textbf{Initial Decision} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.3}$$

Now the Central Node will assign channels to users as per the above decision matrix. After transmitting for T instants, the Central Node will be updated the erroneous estimate of the cost Values of the assigned Channel to users. For example, here, channel 1 was assigned to user 2. The user will report its cost Value with channel 1 to the Central Node after transmitting T instants. Similarly, all users will report to the Central Node after T instants. These values will be updated to the cost Matrix before making the next decision.

$$\textbf{Updated Cost} = \begin{bmatrix} 0 & 7.02 & 0 & 0 \\ 5.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5.06 \\ 0 & 0 & 7.44 & 0 \end{bmatrix} \tag{4.4}$$

When this updated cost Matrix is used to make a new decision, the previously made channel to user assignments would not be decided again. It can be understood from the fact that the networks are trained to obtain a minimum cost, and since all other entries are zero, the subsequent assignments would again depend only on some combination of the network weights. After every T instants, four new entries of the cost Matrix are updated until all the cost Matrix entries are obtained.

$$\begin{aligned} \textbf{Cost} &\longrightarrow \begin{bmatrix} 0 & 7.02 & 3.13 & 0 \\ 5.5 & 8.63 & 0 & 0 \\ 2.83 & 0 & 0 & 5.06 \\ 0 & 0 & 7.44 & 3.2 \end{bmatrix} \\ &\longrightarrow \begin{bmatrix} 0 & 7.02 & 3.13 & 3.71 \\ 5.5 & 8.63 & 2.53 & 0 \\ 2.83 & 8.92 & 0 & 5.06 \\ 6.79 & 0 & 7.44 & 3.2 \end{bmatrix} \\ &\longrightarrow \begin{bmatrix} 4.86 & 7.02 & 3.13 & 3.71 \\ 5.5 & 8.63 & 2.53 & 1.21 \\ 2.83 & 8.92 & 6 & 5.06 \\ 6.79 & 5.66 & 7.44 & 3.2 \end{bmatrix} \end{aligned} \tag{4.5}$$

After a few iterations, the updated cost matrix will be the noisy estimate of the underlying True Covariance Matrix. This measurement can be used to get reliable channel to user assignment. Comparing the assignments using actual underlying cost matrix and its estimate;

$$\textbf{True Cost} = \begin{bmatrix} 4.6 & 7.25 & 3.17 & 3.8 \\ 5.43 & 8.5 & 2.89 & 1.4 \\ 3.06 & 9.1 & 6.06 & 5.18 \\ 6.61 & 5.33 & 7.91 & 3.1 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.6}$$

$$\textbf{Estimated Cost} = \begin{bmatrix} 4.86 & 7.02 & 3.13 & 3.71 \\ 5.5 & 8.63 & 2.53 & 1.21 \\ 2.83 & 8.92 & 6 & 5.06 \\ 6.79 & 5.66 & 7.44 & 3.2 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.7}$$

4.3 Collision avoidance

4.3.1 Greedy Collision Avoidance

As mentioned earlier, since the approach involves multiple pre-trained networks predicting each column of the decision matrix and thereby obtaining the complete decision matrix, there are chances that more than one user would be give access to a particular vacant channel. This is known as a collision. When the accuracy of each Network in the decision prediction system attains an accuracy of 100%, the collision event becomes very rare. This is the case when the number of users and number of channels is less. In such scenarios, finding a heuristic solution rather than the optimal solution becomes an efficient strategy. A low complexity heuristic Algorithm that could be employed here is the greedy collision avoidance Algorithm. When a channel indexed i is assigned to users j_1 and j_2 simultaneously, the underlying costs of the channels with respective users are compared. If $C_{ij_1} < C_{ij_2}$, then the channel i is assigned to user j_1.

Example

Let the cost matrix be as following for a 4 channel, 4 user network.

$$\textbf{Cost}_{4\times 4} = \begin{bmatrix} 9.783 & 7.563 & 1.459 & 1.586 \\ \boxed{5.408} & \boxed{4.181} & 2.143 & 1.662 \\ 6.889 & 6.109 & 8.501 & 1.68 \\ 9.677 & 9.75 & 0.607 & 1.242 \end{bmatrix} \tag{4.8}$$

Using the cost presented to the Network, it makes predictions of individual users, hence estimating each column of the decision matrix

$$\textbf{Initial Prediction}_{4\times4} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.9}$$

Here, it can be seen that channel 2 is simultaneously being assigned to users 1 and 2, and channel 3 is left unassigned. The greedy collision avoidance rule dictates that channel 2 be provided to user 2 as the cost value is lesser compared to user 1, and channel 3 be allocated to user 1.

$$\textbf{Initial Prediction}_{4\times4} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.10}$$

4.3.2 Active Collision Avoidance

When the number of users and number of channels is considerable, this heuristic approach fails to be close to the optimal assignment. As the number of users and channels increases, the dimension of the Convolutional Neural Networks required in each user's decision making increases. Increased size demands a deeper network architecture with an inevitable reduction in the accuracy. Lower accuracy in deciding each column of the decision Matrix causes multiple Networks to make erroneous decisions and cause even more collisions. Applying heuristic greedy collision avoidance techniques would not be an efficient approach since it will affect the overall performance of the assignment technique. When a channel is assigned to more than two users, some channels would be left unassigned. Assigning the multiple unassigned channels to the users who were given the same channel would again be an assignment problem. So, as part of the collision avoidance routine, the assignment is done again for the collided channels and the unassigned channels.

Here, the channel is assigned to the user with the least cost and the remaining users are retained for the subsequent collision avoidance routine. For the collision avoidance assignment, for all valid assignments (not necessarily the correct assignment, here valid means one channel is assigned to a single user), the cost values in the cost matrix are replaced by random numbers of higher magnitudes. It is done to prevent the assignment of these channels and users again by giving a better chance to the unassigned channels. For the channels that had collisions, the user with the least cost is assigned the channel and would be considered a valid assignment; all the other users who were not assigned the channel would retain their cost value. The updated cost matrix for collision Avoidance is then fed as input to the same CNN network. The output at the unassigned locations is observed and updated at the initial decision Matrix. If a collision occurs again, the same procedure is repeated.

Example

$$C_{8\times8} = \begin{bmatrix} 5.091 & 5.592 & 9.82 & 9.301 & 6.251 & 7.825 & 7.47 & 3.567 \\ 3.886 & 3.624 & 2.763 & 1.697 & 4.23 & 8.172 & 5.283 & 5.471 \\ 7.665 & 8.24 & 0.31 & 2.519 & 5.488 & 9.868 & 5.02 & 4.383 \\ 1.335 & 1.544 & 0.993 & 6.002 & 4.249 & 8.872 & 7.058 & 0.349 \\ 1.09 & 7.888 & 5.214 & 6.208 & 4.424 & 3.308 & 8.813 & 0.258 \\ 0.48 & 0.644 & 9.177 & 5.699 & 2.663 & 2.404 & 9.905 & 7.671 \\ 1.494 & 2.962 & 4.212 & 1.327 & 5.475 & 3.658 & 9.475 & 4.366 \\ 6.903 & 1.208 & 0.936 & 8.024 & 1.127 & 0.942 & 1.955 & 8.376 \end{bmatrix} \tag{4.11}$$

Initial decision made by the Neural Networks might not be valid due to the decreased accuracy levels. This is a worst-case example where 4 of the users are assigned the same Channel.

$$\textbf{Network Output}_{8\times8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.12}$$

Here the channel 6 is being assigned to users $1, 2, 5$, and 6. These users need to be reassigned to unassigned channels. Channels 1,4, and 5 are not assigned to any users. Channel 6, which is assigned to multiple users (channel 6), would be assigned to the user who has the least cost (user 1, $C_{61} = 0.48$).

$$\textbf{Updated Decision}_{8\times8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.13}$$

Once the Channel assigned to multiple users is fixed, the channels that were unassigned and the users assigned to a single channel will go in for the collision avoidance routine.

$$\textbf{Modified Cost}_{8\times8} = \begin{bmatrix} 270 & \mathbf{5.592} & 195 & 178 & \mathbf{6.251} & \mathbf{7.825} & 289 & 124 \\ 265 & 115 & 179 & 298 & 212 & 131 & 185 & 112 \\ 210 & 273 & 109 & 224 & 182 & 159 & 188 & 202 \\ 288 & \mathbf{1.544} & 284 & 290 & \mathbf{4.249} & \mathbf{8.872} & 103 & 146 \\ 284 & 203 & 176 & 162 & 131 & 158 & 227 & 291 \\ 101 & 290 & 295 & 139 & 161 & 122 & 241 & 143 \\ 295 & \mathbf{2.962} & 131 & 280 & \mathbf{5.475} & \mathbf{3.658} & 123 & 225 \\ 133 & 145 & 164 & 218 & 230 & 219 & 251 & 101 \end{bmatrix} \tag{4.14}$$

$$\textbf{Collision Decision}_{8\times 8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

These decisions picked from the Collision Avoidance strategy corresponding to the unassigned channels and mis-assigned users are updated at the actual decision Matrix.

$$\textbf{Final Decision}_{8\times 8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.16}$$

The total cost is minimised by using this approach.

Minimum cost using Hungarian Algorithm	$= 15.913$
Total cost using Neural Network with Collision avoidance	$= 19.73$
Worst case Assignment cost using Hungarian Algorithm	$= 64.98$

4.4 Unbalanced Cost Matrix

The cost matrix C is termed balanced when the number of users and the number of jobs are the same. In the Cognitive Radio case, the number of secondary users and the number of channels available for opportunistic spectrum access will not be necessarily the same. Such a system is said to be unbalanced. Here, either the number of channels available would be more than the number of users waiting to be serviced, or the number of SUs is greater than the number of existing spectrum holes. In both cases, zero-padding is done, and the matrix dimension is made equal to the network dimensions.

When the number of users is less than the number of channels available, the assignment procedure is straightforward since the decision-generating system's architecture is such that each Neural Network decides one specific user at a time. After the padded cost matrix is fed as input to the system, only the output vectors corresponding to the available users are observed. A separate architecture could have been employed in this case, using the rectangular matrix as input to find the channel allocation decision. In this work, padding is done, and the cost matrix is made square so that the same architecture could be used for all three cases; the number of users and the number of channels available are equal, greater and lesser.

4.4.1 Example

Consider the cost matrix:

$$C_{6\times 4} = \begin{bmatrix} 1.198 & 3.239 & 2.687 & 9.903 \\ 9.25 & 8.125 & 1.135 & 4.509 \\ 2.923 & 3.014 & 6.802 & 7.592 \\ 6.888 & 7.5 & 4.196 & 5.973 \\ 8.877 & 6.361 & 8.039 & 2.065 \\ 4.544 & 9.954 & 1.495 & 8.73 \end{bmatrix} \tag{4.17}$$

In this example, the number of channels available is less than the number of users waiting, hence the rectangular cost matrix. The cost matrix is converted to a square matrix by zero-padding it to load the cost matrix to the available architecture.

$$C\_pad_{6\times 6} = \begin{bmatrix} 1.198 & 3.239 & 2.687 & 9.903 & 0 & 0 \\ 9.25 & 8.125 & 1.135 & 4.509 & 0 & 0 \\ 2.923 & 3.014 & 6.802 & 7.592 & 0 & 0 \\ 6.888 & 7.5 & 4.196 & 5.973 & 0 & 0 \\ 8.877 & 6.361 & 8.039 & 2.065 & 0 & 0 \\ 4.544 & 9.954 & 1.495 & 8.73 & 0 & 0 \end{bmatrix} \tag{4.18}$$

The system will assign channels to users as if two more users were waiting for spectrum access.

$$\textbf{Network Output}_{6\times 6} = \left(\begin{array}{cccc|cc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right) \tag{4.19}$$

When the decision from the network is obtained, the last two columns of the decision matrix are discarded, and users are given access to the channel based on the first four columns. It can also be observed that channels 4 and 6 are not allocated here.

4.4.2 Fairness in channel assignment

When the number of users exceeds the number of channels available, the assignment procedure is similar to the previous method for most parts. However, unlike previous cases, some users who were not assigned any channel might remain unassigned for the entire transmission duration. It does not necessarily mean that the user has high costs with all channels. The user can have low costs with channels and still be left unassigned because assigning a channel to this unassigned user might increase the total cost of the assignment. Fairness is also one of the critical performance criteria in all resource allocation schemes. The importance of fairness in wireless networks and cognitive radio is studied in [28].

In order to accommodate the users who were left out by the assignment framework, time-sharing over data-frame transmissions is suggested. An algorithm outlining the approach to accommodate the unassigned channels is shown in ALGORITHM 2. The strategy allows the unassigned users to transmit for a fixed fraction of data transmission duration. The aim here is to let the unassigned user get the best channel for a limited time to make necessary communication. At the same time, the total cost of the assignment does not degrade significantly due to the time-sharing.

Algorithm 2 Fairness in channel assignment

1: **Generate Decision:** Channel assignment is obtained using the decision generation framework introduced earlier.
2: **while** cost matrix remains unchanged **do**
3:     **for** each unassigned user **do**
4:         User can access channel for $x\%$ of total transmission time
5:         Select the channel with least cost from the cost matrix: κ
▷ This channel will be used by the unassigned user
6:         Wait for the user accessing the κth channel to transmit $100 - x$ frames.
7:         Channel κ is accessed by the unassigned channel for transmitting x frames.
8:         Channel access is given back to the assigned user.

Example

The initial assignment procedure is similar to when the number of users was less than the number of channels available.

$$C_{5\times 6} = \begin{bmatrix} 9.041 & 6.219 & 8.678 & 5.628 & 3.576 & 8.769 \\ 5.625 & 4.966 & 5.845 & 2.086 & 8.622 & 7.145 \\ 2.748 & 1.868 & 0.809 & 9.213 & 7.277 & 9.021 \\ 8.437 & 1.463 & 3.359 & 4.742 & 2.327 & 8.062 \\ 9.703 & 4.417 & 1.499 & 7.299 & 9.133 & 4.711 \end{bmatrix} \tag{4.20}$$

$$C\_pad_{6\times 6} = \begin{bmatrix} 9.041 & 6.219 & 8.678 & 5.628 & 3.576 & 8.769 \\ 5.625 & 4.966 & 5.845 & 2.086 & 8.622 & 7.145 \\ 2.748 & 1.868 & 0.809 & 9.213 & 7.277 & 9.021 \\ 8.437 & 1.463 & 3.359 & 4.742 & 2.327 & 8.062 \\ 9.703 & 4.417 & 1.499 & 7.299 & 9.133 & 4.711 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.21}$$

$$\textbf{Network Output}_{6\times 6} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.22}$$

Here, fair treatment for user 6 should be incorporated without causing much degradation to the overall cost of the assignment. Time-sharing over transmissions is the only way to provide service to all users in this band limited system. All the cost entries of the user 6 of this particular example, are higher compared to the assigned user to channel costs (see Eq.4.23). The algorithm let the user 6 access a channel for, say, 10% duration of data transmission. During this duration, the algorithm tries to provide the best performance available to the user rather than optimizing the total cost. In this example, the least cost occurs for channel 5, which was assigned to user 3. So now, user 3 will transmit for 90% and user 6 will transmit for 10% of transmission duration

$$\begin{bmatrix} 9.041 & 6.219 & 8.678 & 5.628 & \boxed{3.576} & 8.769 \\ 5.625 & 4.966 & 5.845 & \boxed{2.086} & 8.622 & 7.145 \\ \boxed{2.748} & 1.868 & 0.809 & 9.213 & 7.277 & 9.021 \\ 8.437 & \boxed{1.463} & 3.359 & 4.742 & 2.327 & 8.062 \\ 9.703 & 4.417 & \boxed{1.499} & 7.299 & 9.133 & \boxed{4.711} \end{bmatrix} \tag{4.23}$$

The total cost obtained by assigning channels decision generation network without considering the fairness in the assignment algorithm was **11.372**. When the fairness strategy is applied, the total cost will be higher than the Hungarian lower cost. Since the approach involves time-sharing between users, the total cost is found using the weighted sum of costs for user 3 and user 6.

$$\begin{aligned} \textbf{Total Cost} =& 3.576 + 2.086 + 2.748 + 1.463 \\ & + (0.9 \times 1.499 + 0.1 \times 4.711) \\ =& 11.6932 \end{aligned} \tag{4.24}$$

Chapter 5

Simulation and Results

5.1 Simulation Setup

The dataset required for the training of the Neural network for mimicking the Hungarian algorithm is generated using a python implementation of the Hungarian/Munkres algorithm available at [29]. The input cost matrix is randomly generated between 0 and 10 with a precision of up to 3 decimal points. Suppose the cost matrix entries are of different magnitude range during the deployment phase. In that case, it could be normalized to be between 0 and 10, and the assignment decision can be evaluated using the normalized cost matrix. Two neural network architectures were used for simulation; CNN network with 6 layers was used for cost matrix of lower dimension (4,5,6), and for matrix dimensions 8 and 16, the CNN had 8 layers.

Network Architecture 1: A CNN is constructed with one input layer, two convolutional layers with 4 and 8 filters and a kernel size [1 x 1] each, three fully connected layers with 128, 256 and 64 nodes each and an output layer with nodes equal to the number of channels.

Network Architecture 2: This network consists of one input layer, three convolutional layers with 10, 16 and 32 filters, four fully connected layers with 256, 512, 128 and 64 nodes and an output layer with the number of nodes equal to the number of channels.

The convolutional layers has kernel sizes [2 x 1], [1 x 2] and [2 x 1] respectively.

In both network architectures, the activation used was Rectified Linear Unit *(ReLu)* as it provided better performance than *tanh* and *sigmoid* activation functions [30].The optimizer used was adaptive momentum estimation*(Adam)* which is based on adaptive estimates of lower-order moments[31], and the loss function used is categorical cross-entropy. The learning rate was fixed at 0.0001 to ensure the convergence and stability of the CNNs [27]. All the convolution layer kernels were initialized using the glorot uniform kernel initializer. L2 Regularization was used to penalize layer parameters during training. The regularisation parameter was set to 0.02.

For comparison, the assignment is compared against random assignment and hill-climbing technique. In [20], the hill-climbing technique is used to assign vacant channels to secondary users. In this method, the channels are randomly assigned to the users initially. Then two assignments are randomly swapped to check if the swap results in a better total cost. If swapping does not improve the total cost after n number of swaps, the last configuration is fixed. Here n is the number of channels. If a swap improves the total cost, the swapped configuration is retained, and n more consecutive swaps are done along with a cost comparison with previous costs. This Algorithm is repeated until no swaps result in an improvement in total cost.

5.2 Performance Metrics

Accuracy: Accuracy in channel assignment here is defined as how close can the proposed architecture predict the assignment decision to the Hungarian minimum decision. If all the CNNs predict the correct decisions, then all the columns of the decision matrix will be the same as the columns of the Hungarian decision matrix. In that case, the accuracy would be 100%. If none of the CNNs predicts the correct channel assignment decision, the accuracy would be 0%. Therefore, the overall accuracy of the deciding system depends on the accuracy of individual CNN accuracies.

Percentage Deviation: Even though accuracy is a good performance measure, in this particular problem, an assignment can have an accuracy of 0% and still have a total cost very close to the Hungarian minimum cost. Consider the carefully selected example.

$$\begin{bmatrix} \boxed{2.58} & 2.61 & 3.986 & 8.835 & 9.945 \\ 3.327 & 8.176 & 2.148 & \boxed{2.152} & 4.481 \\ 2.7 & 4.091 & 5.112 & 2.836 & \boxed{2.604} \\ 5.6 & \boxed{0.521} & 6.62 & 0.31 & 1.964 \\ 4.888 & 3.45 & \boxed{2.705} & 2.773 & 3.448 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.1}$$

$$\textbf{Total cost} = 10.562$$

When the decision generation network makes an assignment decision as in the equation 5.1, the accuracy would be 100%. The cost of such an assignment is 10.562. Now consider another assignment for the same cost matrix (Eq. 5.2).

$$\begin{bmatrix} 2.58 & \boxed{2.61} & 3.986 & 8.835 & 9.945 \\ 3.327 & 8.176 & \boxed{2.148} & 2.152 & 4.481 \\ \boxed{2.7} & 4.091 & 5.112 & 2.836 & 2.604 \\ 5.6 & 0.521 & 6.62 & \boxed{0.31} & 1.964 \\ 4.888 & 3.45 & 2.705 & 2.773 & \boxed{3.448} \end{bmatrix} \Longrightarrow \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

$$\textbf{Total cost} = 11.216$$

This assignment is not the optimal assignment for the cost matrix. However, the cost of this assignment is 11.216, which is very close to the optimal assignment cost. If accuracy was used to measure the performance, this assignment would have a 0% accuracy. So, in this work, another metric that measures the closeness of an assigned cost to the Hungarian minimum cost is used.

The total cost of an assignment is compared with the lowest possible total cost and worst-case total cost to get the percentage deviation from the best cost (minimum cost). For a given cost matrix, if the assignment results in the Hungarian minimum cost, then the percentage deviation from best cost would be zero. If the assignment results in the Hungarian maximum cost, it is represented as a 100% deviation from the best cost. Since the cost matrix entries are from a uniform distribution, using random assignment would have a percentage deviation of 50.

In the example, the worst case assignment would have a total cost of 32.47. Therefore, the assignment in Eq. 5.2 has a percentage deviation of 2.99% from the best case cost.

5.3 Results

The performance of the proposed architecture was evaluated for when the number of users and number of channels is equal to 4, 5, 6, 8 and 16. The overall decision accuracy depended on the accuracy of individual CNNs. The average test set accuracy of each CNN for different number of users is recorded in Table 5.1.

Number of users	4	5	6	8	16
Accuracy	96.4%	88.3%	71.67%	63.61%	54.9%

Table 5.1: Average accuracy of CNNs vs Number of users

The overall accuracy will be much less than the individual Neural network accuracies as the error from each network combine. This effect is reduced by using Active collision avoidance after obtaining the individual outputs from each CNN. The overall performance figures of the decision generation framework are listed in comparison with random assignment, and hill-climbing is listed in Table 5.2.

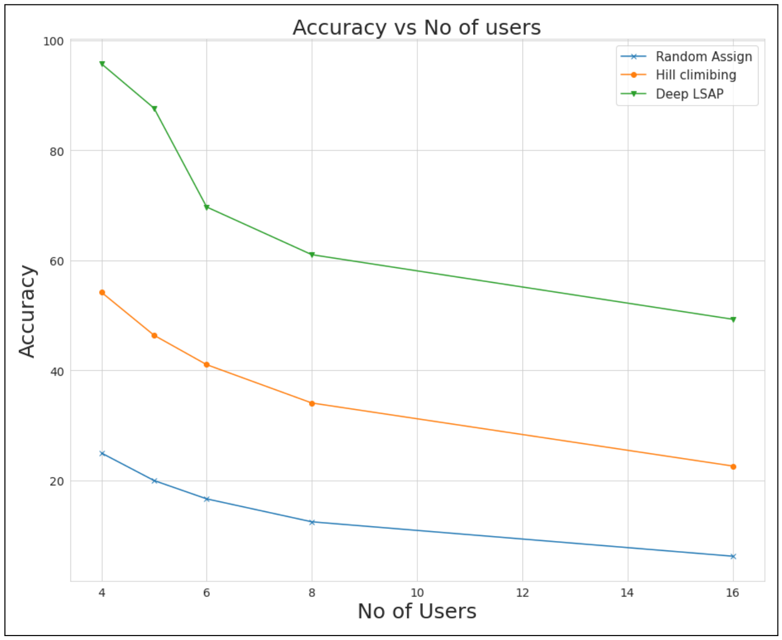

Figure 5.1: Accuracy comparison of Assignment techniques

In Figure 5.1, the accuracy in decision making is plotted against the number of users. It can be observed that as the number of users increases, the accuracy decreases. The decline comes from the fact that as the number of users increases, the dimension of convolutional layers should be increased, and the network needs to be made deeper. Achieving higher accuracy for the case when users are more comes with the expense of making the decision generation slower.

Accuracy					
Number of Users	4	5	6	8	16
Random Assign	25%	20%	16.67%	12.5%	6.25%
Hill Climbing	54.21%	46.4%	41.06%	34.09%	22.64%
Deep LSAP	**95.73%**	**87.66%**	**69.71%**	**61.04%**	**49.31%**
Percentage Deviation from the optimal Cost					
Hill Climbing	17.98	18.56	18.72	18.96	16.3
Deep LSAP	**0.42**	**3.66**	**4.9**	**6.44**	**9.8**

Table 5.2: Performance vs dimension of cost matrix

In Tables 5.3 and 5.4, the performance of unbalanced cost matrix is analysed for the case when number of channels available is 6 and 8 respectively. The number of users is varied from 3 to the number of channels. It can be observed that when the number of users is less, the accuracy is more and when the number of users is increased until the cost matrix becomes balanced, the accuracy decreases and approaches the corresponding accuracy recorded in Table 5.2.

Number of users	3	4	5	6
Accuracy	89.21	82.75	75.81	71.96

Table 5.3: Performance of unbalanced assignment (6 channels)

Number of users	3	4	5	6	7	8
Accuracy	90.41	85.03	79.88	73.9	66.36	63.97

Table 5.4: Performance of unbalanced assignment (8 channels)

Chapter 6

Conclusions and Future Work

As the demand for wireless spectrum is increased with more and more devices getting connected to cloud and IoT with the emergence of 5G, cognitive radio has become significant now more than ever. However, while cognitive radio promises to efficiently use the communication spectrum by utilizing the underutilized portion of the spectrum, the detected spectral holes are not so efficiently redistributed among the secondary users. In this thesis, a scheme that assigns channels to users based on the cost information present at the central node is proposed. Deep learning is used to make the optimal assignment faster as compared to conventional methods. When the number of users outnumbered the number of channels, some users will not get a channel. Fairness was incorporated in the assignment strategy so that no user was denied access to opportunistic communication. Developing an algorithm to make the fairness in assignment more optimal than a fixed strategy would be an exciting extension of this work. Similarly, adding a parameter for monitoring the change in the cost matrix from time to time would also be a worthy extension.

REFERENCES

[1] Spectrum Efficiency Working Group. Report of the spectrum efficiency working group, fcc, 2002.

[2] J. Mitola and G.Q. Maguire. Cognitive radio: making software radios more personal. *IEEE Personal Communications*, 6(4):13–18, 1999.

[3] C. Cordeiro, K. Challapali, D. Birru, and Sai Shankar. Ieee 802.22: the first worldwide wireless standard based on cognitive radios. In *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, pages 328–337, 2005.

[4] Ian F Akyildiz, Won-Yeol Lee, Mehmet C Vuran, and Shantidev Mohanty. A survey on spectrum management in cognitive radio networks. *IEEE Communications magazine*, 46(4):40–48, 2008.

[5] D. Cabric, S.M. Mishra, and R.W. Brodersen. Implementation issues in spectrum sensing for cognitive radios. In *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*, volume 1, pages 772–776 Vol.1, 2004.

[6] Kwang-Cheng Chen and Ramjee Prasad. *Cognitive radio networks*. John Wiley & Sons, 2009.

[7] D. Cabric, I.D. O'Donnell, M.S.-W. Chen, and R.W. Brodersen. Spectrum sharing radios. *IEEE Circuits and Systems Magazine*, 6(2):30–45, 2006.

[8] Long Bao Le and Ekram Hossain. Resource allocation for spectrum underlay in cognitive radio networks. *IEEE Transactions on Wireless communications*, 7(12):5306–5315, 2008.

[9] Yasin Yilmaz, Ziyu Guo, and Xiaodong Wang. Sequential joint spectrum sensing and channel estimation for dynamic spectrum access. *IEEE Journal on Selected Areas in Communications*, 32(11):2000–2012, 2014.

[10] Ivan Christian, Sangman Moh, Ilyong Chung, and Jinyi Lee. Spectrum mobility in cognitive radio networks. *IEEE Communications Magazine*, 50(6):114–121, 2012.

[11] J Von Neuman and O Morgenstern. Theory of games and economic behavior, princeton university press, 1947.

[12] Tadilo Endeshaw Bogale, Luc Vandendorpe, and Long Bao Le. Sensing throughput tradeoff for cognitive radio networks with noise variance uncertainty. In *2014 9th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, pages 435–441, 2014.

[13] Khaled Ben Letaief and Wei Zhang. Cooperative communications for cognitive radio networks. *Proceedings of the IEEE*, 97(5):878–893, 2009.

[14] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity. part i. system description. *IEEE Transactions on Communications*, 51(11):1927–1938, 2003.

[15] Andrew Sendonaris, Elza Erkip, and Behnaam Aazhang. User cooperation diversity. part ii. implementation aspects and performance analysis. *IEEE Transactions on communications*, 51(11):1939–1948, 2003.

[16] Yunxue Liu, Dongfeng Yuan, Mingyan Jiang, Hui Yu, Chunyuan Xu, and Pingping Zhao. Cooperative spectrum sensing in cognitive networks. In *TENCON 2008-2008 IEEE Region 10 Conference*, pages 1–6. IEEE, 2008.

[17] Munehiro Matsui, Hiroyuki Shiba, Kazunori Akabane, and Kazuhiro Uehara. A cooperative sensing technique with weighting based on distance between radio stations. In *2008 14th Asia-Pacific Conference on Communications*, pages 1–4, 2008.

[18] Nancy Nayak, Vishnu Raj, and Sheetal Kalyani. Leveraging online learning for css in frugal iot network, 2020.

[19] Jarmo Lundén, Mehul Motani, and H. Vincent Poor. Distributed algorithms for sharing spectrum sensing information in cognitive radio networks. *IEEE Transactions on Wireless Communications*, 14(8):4667–4678, 2015.

[20] Thulasi Tholeti, Vishnu Raj, and Sheetal Kalyani. A centralized multi-stage non-parametric learning algorithm for opportunistic spectrum access, 2020.

[21] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[22] Merrill M Flood et al. On the hitchcock distribution problem. *Pacific Journal of mathematics*, 3(2):369–386, 1953.

[23] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[24] Nobuaki Tomizawa. On some techniques useful for solution of transportation network problems. *Networks*, 1(2):173–194, 1971.

[25] Jack Edmonds and Richard M Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

[26] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys Tutorials*, 21(3):2224–2287, 2019.

[27] Mengyuan Lee, Yuanhao Xiong, Guanding Yu, and Geoffrey Ye Li. Deep neural networks for linear sum assignment problems. *IEEE Wireless Communications Letters*, 7(6):962 – 965, 2018.

[28] Huaizhou SHI, R. Venkatesha Prasad, Ertan Onur, and I.G.M.M. Niemegeers. Fairness in wireless networks:issues, measures and challenges. *IEEE Communications Surveys Tutorials*, 16(1):5–24, 2014.

[29] Munkres implementation for python. `https://software.clapper.org/munkres/index.html`. Accessed: 2010-09-30.

[30] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.