Transfer Learning Based Joint Resource Allocation for Underlay D2D Communications

R. Jaiswal, S. Deshmukh, M. Elnourani, B. Baltasar Lozano

Abstract: In this paper, we investigate the application of transfer learning to train a Deep Neural Network (DNN) model for joint channel and power allocation in underlay D2D communication. Based on the traditional optimization solutions, generating training datasets for scenarios with perfect channel state information (CSI) is not computationally demanding, compared to scenarios with imperfect CSI. Thus, a transfer learning-based approach can be exploited to transfer the DNN model trained for the perfect CSI scenarios to the imperfect CSI scenarios. We also consider the issue of defining the similarity between two types of resource allocation tasks. For this, we first determine the value of outage probability for which two resource allocation tasks are the same, that is, for which our numerical results illustrate the minimal need for relearning from the transferred DNN model. For other values of outage probability, there is a mismatch between the two tasks and our results illustrate a more efficient relearning of the transferred DNN model. Our results show that the learning dataset required for relearning of the transferred DNN model is significantly smaller than the required training dataset for a DNN model without transfer learning.

E.1 Introduction

In the last decade, research efforts in underlay D2D communication have categorically focused on devising judicious resource allocation algorithms, as it is fundamental for the efficient performance of both cellular and D2D networks [84–86]. Judicious resource allocation in the form of prudent power control and channel assignment to D2D pairs is vital to limit interference. Its main goal is to maximize performance metrics such as sum rate, energy efficiency, spectral efficiency etc. while satisfying the desired quality of service (QoS) requirements [84, 86]. Unfortunately, most formulations on resource allocation lead to either mixed-integer nonlinear problems or highly non-convex problems, which in general involve combinatorial complexity for obtaining the optimal solution. Thus, convex relaxation approaches can be exploited to obtain the sub-optimal solution with consideration of (i) optimality gap, (ii) convergence guarantees, and (iii) computation complexity.

In order to practically realize the resource allocation solution, recent research works have proposed training of Deep Neural Networks (DNNs) to a close-to-optimal solution. Motivation for exploiting DNN is primarily due to its universal approximation capability [87] and supplemented by the fact that trained DNN models are computationally very simple [88] to execute. In this context, deep learning-based resource allocation algorithms have been proposed in [89, 90]. However, these deep learning-based resource allocation algorithms are only implementing power control and power allocations, but not the joint resource allocation of both power allocation and channel assignments, as presented in the perfect CSI [70] and imperfect CSI [72] cases. Moreover, these previous works [89, 90] are focused on perfect CSI and imperfect CSI scenario of resource allocation in terms of power control and power allocation respectively but do not include any transfer learning (TL) approach in order to either improve the learning performance or accelerate the training, saving computational resources.

Notice that the success of DNN models in replicating different optimization-based resource solutions relies heavily on the availability of sufficiently large learning datasets. The generation of large datasets depends on the computational complexity of the original resource allocation solution; for example, under the perfect CSI scenario, [70] decouples without loss of optimality, the resource allocation problem into multiple power allocation subproblems and a channel assignment subproblem. In [70], the power allocation subproblems have closed-form solutions and the channel assignment subproblem is solved by integer relaxation. Thus, in this case, it is possible to obtain a large training dataset with reasonable complexity. However, if we consider a similar set-up with imperfect CSI [72], the decoupled power allocation sub-problems are solved iteratively using fractional programming, making the generation of large learning datasets cumbersome work.

The resource allocation under perfect and imperfect CSI can be considered as similar tasks and in this paper, we investigate how to exploit the concept of transfer learning (TL) in order to address the problem of learning with small datasets. Transfer learning [19] is a promising technique in which DNN models trained for one task can be transferred to another similar task which has less learning data. Some related research work on TL includes robust sensing framework [21], and transfer learning via self-imitation for resource allocation [32].

In this work, we address the problem of learning resource allocation in an imperfect CSI scenario [72] by exploiting the TL from the perfect CSI scenario. We first characterize the similarity between the two resource allocation tasks in terms of the outage probability. Next, we train a DNN model under the perfect CSI scenario with the dataset generated from the algorithm presented in [70] and define it as a baseline model for TL. The providing baseline model is then retrained for the case of imperfect CSI with a small dataset generated from the algorithm presented in [72]. Our numerical results illustrate that as the mismatch between two tasks increases, a larger dataset is required for relearning the baseline model; however, the amount of dataset required for relearning the baseline model is significantly smaller, compared to training of the new DNN model.

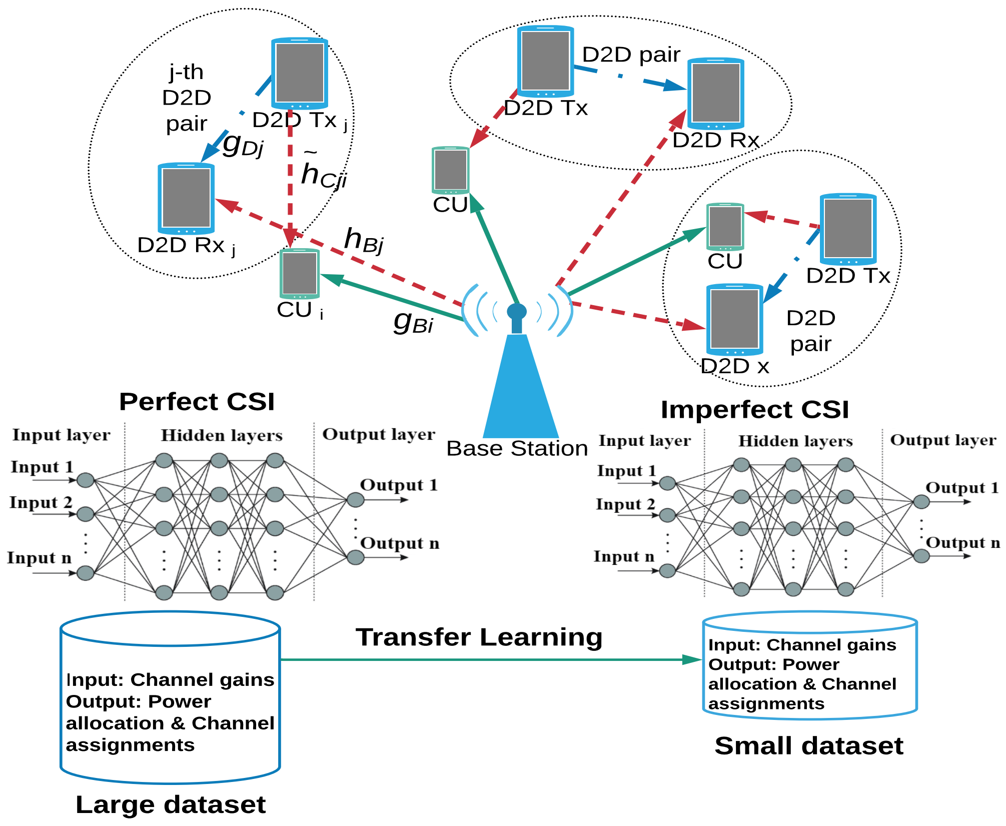

Figure E.1: Illustration of the system model with TL approach.

The remainder of this paper is structured as follows: Section E.2 describes the system model and problem formulation of resource allocation under perfect and imperfect CSI. Section E.3 presents the proposed deep learning and transfer learning-based approach for resource allocation. Section E.4 presents the experimental system setup and Section E.5 presents and discusses the results before the concluding remarks and future directions are provided in Section E.6.

E.2 System Model

In this work, we investigate the transfer of a DNN model trained for resource allocation (channel assignment and power allocation) to cellular users (CUs) and D2D pairs from the case of perfect CSI conditions to the case of practical operations under imperfect CSI conditions, as shown in Figure E.1. Here, we consider a cell enabled by a base station (BS), which communicates with N_C cellular users through N_C downlink channels. The cell is assumed to operate in fully loaded mode; thus, CUs can be indexed by $\mathcal{C} = \{1, ..., N_C\}$. Next, we consider N_D D2D pairs (indexed by $\mathcal{D} = \{1, ..., N_D\}$) wishing to communicate in underlay using the aforementioned N_C downlink channels. The notations for channels are as follows: g_{B_i} and g_{D_j} denote respective direct channel gains between, BS to i-th CU and transmitter and receiver in D2D pair; h_{B_j} and $h_{C_{j,i}}$ denotes respective interference channel gain between BS to the receiver of j-th D2D pair and the transmitter of j-th D2D pair to i-th CU. Further, we denote the total noise power in any channel by N_0.

Let $\beta_{i,j}$ be a binary variable denoting the channel assignment to the j-th D2D pair; $\beta_{i,j} = 1$, if i-th CU shares channel with j-th D2D pair and $\beta_{i,j} = 0$ otherwise. The D2D pairs are allowed to simultaneously access multiple channels; thus, improving their individual sum rate. However, in order to limit interference among D2D pairs, sharing of a channel is restricted to at most one D2D pair, i.e., $\sum_{j=1}^{N_D} \beta_{i,j} \leq 1, \forall i$. Similarly, let P_{B_i} and $P_{D_{j,i}}$ denote respectively transmit power allocated to the BS over the i-th channel and the j-th D2D pair when accessing the i-th channel. The corresponding transmit powers are constrained as: $P_{B_i} \leq P_{B_{\max}}$ and $P_{D_{ji}} \leq P_{D_{\max}}$. Given the above system model, we define the following two resource allocation tasks:

E.2.1 Task I: Resource Allocation under Perfect CSI

In this task, we follow the same formulation as in [70]. They assume that all channel gains g_{B_i}, g_{D_j}, h_{B_j} and $h_{C_{j,i}}$; $1 \leq i \leq N_C$, $1 \leq j \leq N_D$ are perfectly known at the BS. The objective of this task is the sum rate maximization of both the cellular and the D2D network along with a fairness measure in channel assignment to the D2D pairs. Under the assumption of capacity-achieving codes, let $\Gamma(z) := \log_2(1+z)$; the sum rate over i-th channel is defined as: $R_i := \sum_{j \in \mathcal{D}} \beta_{i,j}[R_{C_{i,j}} + R_{D_{j,i}}] + (1 - \sum_{j \in \mathcal{D}} \beta_{i,j}) R_{C_{i,0}}$, where $R_{C_{i,j}} = \Gamma(P_{B_i} g_{B_i}/(N_0 + P_{D_{ji}} h_{C_{j,i}}))$ denotes the rate of the i-th CU when sharing the channel with the j-th D2D pair ($\beta_{ij} = 1$); $R_{D_{j,i}} = \Gamma(P_{D_{ji}} g_{D_j}/(N_0 + P_{B_i} h_{B_j}))$ the rate of the j-th D2D pair when sharing the channel with the i-th CU ($\beta_{ij} = 1$); and $R_{C_{i,0}} = \Gamma(P_{B_{max}} g_{B_i}/N_0)$ the rate of the i-th CU when it shares its channel with no D2D pair ($\beta_{ij} = 0\ \forall j$). The overall network rate of both cellular and D2D networks can be expressed as $R := \sum_{i \in \mathcal{C}} R_i$. For consideration of fairness in channel assignment to D2D pairs, they define unfairness measure (from [71]): $\delta^2(B) := 1/(N_D x_0^2) \sum_{j=1}^{N_D} (x_j(B) - x_0)^2$, where $x_j := \sum_{i=1}^{N_C} \beta_{i,j}$ is the number of channels assigned to the j-th D2D pair, $x_0 := N_C/N_D$ and B is channel assignemnet. Here, if N_C is an integer multiple of N_D, then $x_j = x_0\ \forall j$ is the fairest channel assignment possible. Finally, the overall resource allocation optimization problem under perfect CSI can be formulated as:

$$\max_{B, P_B, P_D} \quad R(B, P_B, P_D) - \gamma \delta^2(B) \tag{E.1a}$$

$$\text{subject to} \quad \beta_{i,j} \in \{0,1\}, \ \sum_{j=1}^{N_D} \beta_{i,j} \leq 1\, \forall i \tag{E.1b}$$

$$0 \leq P_{B_i} \leq P_{B_{max}} \forall i \quad 0 \leq P_{D_{ji}} \leq P_{D_{max}} \forall j, i \tag{E.2a}$$

$$\forall i, j, \ \frac{P_{B_i} g_{Bi}}{N_0 + P_{D_{ji}} h_{C_{j,i}}} \geq \eta^{C_{min}} \text{ if } \beta_{ij} = 1 \tag{E.2b}$$

$$\forall i, j, \ \frac{P_{D_{ji}} g_{Dj}}{N_0 + P_{B_i} h_{B_j}} \geq \eta^{D_{min}} \text{ if } \beta_{ij} = 1. \tag{E.2c}$$

where $\eta^{C_{min}}$ and $\eta^{D_{min}}$ are the respective minimum signal-to-interference plus noise ratio (SINR) requirements for CUs and D2D pairs. Notice that problem (E.1) is a mixed-integer non-convex program, which involves combinatorial complexity for obtaining the optimal solution. A close-to-optimal solution to problem (E.1) is provided in [70], where joint power allocation and channel assignment are optimally decoupled in several power allocation problems and a channel assignment sub-problem. The decoupled power allocation subproblems have closed-form solutions and the channel assignment subproblem is solved by integer relaxation. This solution is computationally efficient. Thus, due to low complexity, one can easily obtain a large dataset to transfer a DNN model.

E.2.2 Task II: Resource Allocation under Imperfect CSI

In this task, we follow the same formulation as in [72]. The interference channel gain from the transmitter of the j-th D2D pair to the i-th CU, i.e., $h_{C_{j,i}}$, is considered to be estimated with minimum cooperation between the cellular and D2D networks. Thus, the interference channel gain is assumed to be exponentially distributed (Rayleigh fading) and is denoted by $\tilde{h}_{C_{j,i}}$; $1 \leq i \leq N_C$, $1 \leq j \leq N_D$. Due to imperfect CSI, the resource allocation optimization problem presented in (E.1) incurs the following modifications: (i) the objective function, and (ii) the stochastic minimum SINR constraint for the CUs.

The stochastic SINR constraint (E.2b) can be replaced with probabilistic constraint to guarantee a minimum outage probability ϵ, expressed as:

$$\Pr\left\{\frac{P_{B_i} g_{B_i}}{N_0 + P_{D_{ji}} \tilde{h}_{C_{j,i}}} \geq \eta^C_{min}\right\} \geq (1-\epsilon) \text{ if } \beta_{ij} = 1,\ \forall i, j \tag{E.3}$$

The probabilistic SINR constraint can be expressed in a closed form expression for a given statistical distribution of $h_{C_{j,i}}$. Thus, constraint (E.3) can be equivalently expressed as:

$$\frac{P_{B_i} g_{B_i}}{N_0 + P_{D_{ji}} F^{-1}_{\tilde{h}_{C_{j,i}}}(1-\epsilon)} \geq \eta^C_{min} \tag{E.4}$$

where, $F^{-1}_{\tilde{h}_{C_{j,i}}}(1-\epsilon)$ is the inverse cumulative distribution function (CDF) for $\tilde{h}_{C_{j,i}}$ evaluated at $(1-\epsilon)$.

Next, focusing on the stochastic objective function, the objective function (E.1a) can be replaced by the criterion to maximize the minimum network rate exceeded for $(1-\epsilon)$ portion of the time. The minimum network rate can be considered by analyzing the lower bound of the total rate at channel i, which is defined as $R_i^{LB} := (1 - \sum_{j\in\mathcal{D}} \beta_{i,j}) R_{C_{i,0}} + \sum_{j\in\mathcal{D}} \beta_{i,j} [R_{D_{j,i}} + R^{LB}_{C_{i,j}}]$, where, $R^{\mathrm{LB}}_{C_{i,j}}$ denotes the lower bound (which must be at least achieved $(1-\epsilon)$ portion of the time) of the rate of the i-th CU when sharing the channel with the j-th D2D pair ($\beta_{ij} = 1$). Since $\tilde{h}_{C_{j,i}}$ is random, we can compute $R^{LB}_{C_{i,j}} = \Gamma(z^{LB}_{C_{i,j}})$ where $z^{LB}_{C_{i,j}} : \Pr\{z^{LB}_{C_{i,j}} \leq P_{B_i} g_{B_i} / (N_0 + P_{D_{ji}} \tilde{h}_{C_{j,i}})\} = 1-\epsilon$. The minimum sum rate is therefore $R(B, P_B, P_D) := \sum_{i\in\mathcal{C}} R_i^{LB}$. The fairness part is the same.

Once again, note that the resource allocation problem (E.1) with minimum network rate objective and minimum SINR constraint for CUs expressed in (E.4), is a non-convex mixed-integer program and obtaining the optimal solution requires combinatorial complexity. An efficient close-to-optimal solution to the resource allocation problem for the case of imperfect CSI is provided in [72], where once again the joint power allocation and channel assignment problem is decoupled, without loss of optimality, in several power allocation and channel assignment subproblems. The decoupled power allocation subproblems are solved using iterative fractional programming, whose solutions are later used to obtain channel assignment by integer relaxation. Notice that due to the iterative fractional programming in solving the power allocation sub-problem, generating large datasets to train a DNN is cumbersome work due to larger computational complexity. Thus, in the next section, we explore the similarity between these two tasks: (i) resource allocation in the perfect CSI case, and (ii) resource allocation in the imperfect CSI case to exploit transfer learning.

E.3 Transfer Learning based Resource Allocation

The fundamental requirement in transfer learning is to establish a similarity between the tasks over which TL is performed. For the presented resource allocation problem, the similarity between two tasks: (i) resource allocation in perfect CSI and (ii) resource allocation in imperfect CSI can be established by following Lemma.

Lemma E.3.1. *Under the assumption that the interference channel gains $\tilde{h}_{C_{j,i}}$ follows an exponential distribution with the mean denoting the true channel gain value in the perfect CSI case, the two resource allocation tasks for perfect CSI and imperfect CSI coincide for outage probability $\epsilon = \frac{1}{e}$.*

Proof: Notice that the minimum network rate objective function with the modified constraint (E.4) for resource allocation in the imperfect CSI case, is the same as for the perfect CSI case when $F^{-1}_{\tilde{h}_{C_{j,i}}}(1-\epsilon) = h_{C_{j,i}}$. For exponential distribution, $F^{-1}_{\tilde{h}_{C_{j,i}}}(1-\epsilon) = \mathbb{E}[\tilde{h}_{C_{j,i}}]\ln\left(\frac{1}{\epsilon}\right)$. Since the true CSI is equal to the mean of $\tilde{h}_{C_{j,i}}$, i.e., $h_{C_{j,i}} = \mathbb{E}[\tilde{h}_{C_{j,i}}]$, the two tasks are the same for $\epsilon = \frac{1}{e} = 0.3679$.
Next, we briefly discuss the DNN architecture, which is used in this work to obtain the resource allocation solution for the perfect CSI case.

E.3.1 Fully-connected DNN Architecture

A deep neural network (DNN) architecture consists of multiple numbers of layers between the input and output, each of which consists of a linear operation followed by a point-wise non-linearity, also known as the activation function.

Consider a feed-forward DNN with L layers, labelled $l = 1, ..., L$ and each with a corresponding dimension q_l, as shown in Figure E.2. The layer l is defined by the linear operation $\mathbf{W_l} \in \mathbb{R}^{q_l - 1 \times q_l}$ followed by a non-linear activation function $\sigma_l : R^{q_l} \to R^{q_l}$. Layer l receives input from the $l-1$ layer denoted as, $\mathbf{W_{l-1}} \in R^{q_{l-1}}$,

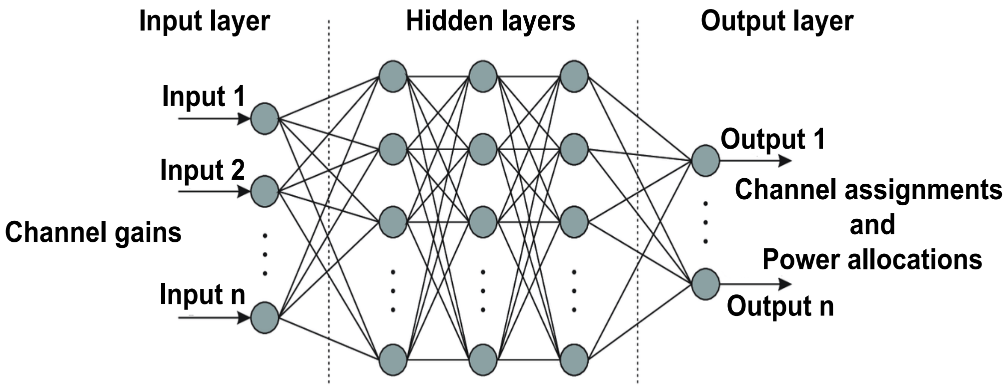

Figure E.2: Architecture of fully-connected DNN with input and output.

the resulting output of the layer l, $W_l \in R^{q_l}$, is then computed as $\mathbf{W_l} := \sigma_l(W_l\mathbf{W_{l-1}})$, where $\sigma_l(\cdot)$ is point-wise activation function. The final output of the DNN, W_L, is then related to the input W_0 by propagating through the various layers of the DNN as $W_L = \sigma_L(W_L(\sigma_{L-1}(W_{L-1}(....(\sigma_1(W_1W_0))))))$. The DNN learns the layer-wise weights $W_1, W_2, ..., W_L$. Our input to DNN is channel gains and output is channel assignments and power allocations. The activation function σ_l includes a rectifier function (commonly referred to as ReLU), defined as $\sigma_l(x) = 0$ for $x < 0$ and x for $x > 0$.

E.3.2 Output Discretization and Scaling

In order to satisfy constraints (E.1b) and (E.2a), the output vector in the resource allocation problem, that is, the channel assignments and power allocations, are discretized and scaled respectively. A binary channel assignment to j-th D2D pair is assigned as 1 if the channel is assigned and 0 if the channel is not assigned. Since each channel can be assigned to at most one of the N_D D2D pairs, for a j-th D2D pair, we discretize maximum among N_C possible channel assignments values $\{\beta_{i,j}\}_{i\in\mathcal{C}}$ to one and other assignments to zero. Similarly, the power value should range between 0 and $P_{\max}$. However, the obtained output power values may be higher than $P_{\max}$ and lesser than 0. Thus, in order to bring it to the specified range, the power value is thresholded with 1 such that if the power is more than 1, then it will be mapped to 1 and if it is less than 0, then it is mapped to 0, followed by multiplication by $P_{\max}$.

E.3.3 Transfer Learning Strategy

In transfer learning, a DNN model trained for some specified task is transferred to perform a similar task decreasing substantially the retraining [91]. A given target task can be trained with fewer data by taking the trained model of a similar task as the baseline model. Here the trained baseline DNN model refers to the model trained for resource allocation in the perfect CSI scenario and the target task is resource allocation for the imperfect CSI scenario.

Thus, for this work, we train a baseline model for the perfect CSI case using a large dataset generated by the computationally efficient algorithm presented in [70]. Next, we use this baseline model to initialise the weights of the DNN model and fine-tune it for the imperfect CSI scenario, where we assume that we have an insufficient amount of dataset (generated from [72]) for training and testing, due to the huge complexity of the considered algorithm.

E.4 System Setup

The proposed DNN and TL-based approaches are implemented in Python 3.7.3 with TensorFlow 2.2.0 on a Windows 10 laptop having an Intel Core i5 8^{th} generation processor, Intel UHD Graphics 620, and 16 GB of memory. The original solver for the perfect and the imperfect CSI model of resource allocation is implemented in MATLAB R2018a. Therefore, we have compared the computational performance of the perfect and imperfect CSI model with the proposed DNN and TL-based approach.

E.4.1 Data Generation

The simulation setup to generate data comprises a circular cell of a 500m radius in which the CUs and D2D transmitters are placed uniformly at random. Each D2D receiver is placed uniformly at random inside a circle of radius 5m centred at the corresponding transmitter. The channel gains are calculated using a path-loss model with exponent 2 and gain -5 dB at a reference distance of 1m. We assume $\tilde{h}_C$ to be exponentially distributed with the mean value obtained from the mentioned path-loss model. Averages over 100,000 independent realizations of the user locations with parameters $BW = 15$ kHz, $\gamma = 50 \times BW$, $N_D = 5$, $N_C = 5$, $N_0 = -70$dBW (γ is scaled with BW to ensure that the unfairness and the achieved rate are of comparable values) are performed. Thus, the input to the DNN is the set of channel gains, that accounts for 40 inputs. The output to the DNN is a joint set of 50 power allocations and 25 channel assignment variables. For training the baseline model with perfect CSI, we consider 100,000 input-output pair samples.

E.4.2 Parameter Selection

To obtain a satisfactory baseline model, we train DNNs with a different number of hidden layers, each having a variable number of neurons. Since output values (channel assignments and power allocations) are positive, the activation function to the hidden layers is a rectified linear unit (ReLU) and the output layer is linear. The weights of the baseline DNN are initialized randomly. We use mean square error (MSE) as the loss function and ADAM optimizer with a learning rate of 0.001 for stochastic optimization. We standardize the dataset by taking the mean and scaling to unit variance. We use a mini-batch of 256 samples. Training epochs are set empirically.

E.4.3 Training and Testing Stage

For the baseline DNN training, we divide the whole data into 80:20 ratio, that is, 80 % for training and 20 % for testing. During testing, for each channel realization, we pass it through the trained network and collect the optimized power allocation and channel assignments. Then, we also evaluate the satisfaction of power constraints (E.2b), SINR (E.2c), and (E.4).

E.5 Results and Discussions

This section presents numerical results and a discussion to showcase the effectiveness of the proposed DNN and TL-based approach. The performance accuracy of the model is evaluated in terms of MSE. There are 100,000 input-output samples in both scenarios.

Table E.1 presents the different training and testing errors that are carried out for the perfect CSI scenario to train the DNN model and set it as our baseline model for TL. It can be observed that the DNN model having 3 hidden layers (highlighted) is performing better as compared to the other combinations. The accuracy, i.e., test MSE, are comparable to the training part, thus illustrating no over-fitting in the model. Moreover, the percentage of constraint satisfaction (E.2b) and (E.2c) in the testing stage are high and nearly the same as its counterpart in training.

Table E.2 shows the results for the imperfect CSI scenario with outage probability $\epsilon = 0.1$ when (*i*) trained via a DNN model with the same training-testing split and configurations as in the perfect CSI scenario, and (*ii*) trained with a baseline model learnt from perfect CSI (TL) with 50% training-testing split. It can be noticed that the TL-based approach has obtained nearly the same MSE as the direct DNN-based approach with the random initialization, and has a similar percentage of constraint satisfaction. This indicates that our baseline model obtained/transferred from the

Table E.1: Model learning for Perfect CSI

Number of hidden layers	**3**	6	3	6
Input layer neurons	**40**	40	40	40
Hidden layer activation function	**ReLU**	ReLU	ReLU	ReLU
Hidden layer neurons	**40,20,5**	40,20,5, 20,40,60	40,60,80	40,60,80, 100,80,60
Output layer activation function	**Linear**	Linear	Linear	Linear
Output layer neurons	**75**	75	75	75
Test MSE	**0.038**	0.039	0.205	0.090
Train MSE	**0.039**	0.039	0.206	0.090
Test E.2b; E.2c constraints satisfied (in %)	**94.53; 100**	94.69; 100	81.78; 86.70	84.67; 92.96
Train E.2b; E.2c constraints satisfied (in %)	**94.55; 100**	94.58; 99.99	81.78; 86.62	84.53; 92.93

Table E.2: Model learning for Imperfect CSI

	Test MSE	Train MSE	Test E.2b constraint satisfied	Train E.2b constraint satisfied	Test E.4 constraint satisfied	Train E.4 constraint satisfied
DNN	0.017	0.017	95.28	95.32	100.0	100.0
TL	0.017	0.017	95.01	94.92	100.0	100.0

perfect CSI scenario improves the learning for the imperfect CSI case with less amount of training data (usually termed as sample-complexity).

The amount of training data required to retrain the baseline model depends on the degree of task mismatch. It can be observed from Figure E.3 that when two tasks are similar, i.e., outage probability $\epsilon = 0.3679$, then just with 20% of training data, the MSE of TL is 0.04. Moreover, with 30% of training data, the transferred baseline model is completely trained. However, as illustrated in Figure E.4, when a lot of mismatch presents between the two tasks, $\epsilon = 0.05$, then almost 80% of training data is required for retraining the transferred baseline model. Since the problem is highly non-convex, the solution that is achieved may correspond to local minima. In Figure E.4, the training of the DNN seems to converge to a local optima when the amount of training data is below 70%, but then it achieves a better solution with 80% of the training data.

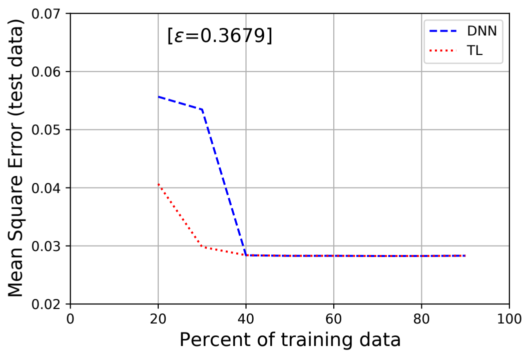

Figure E.3: Percent of training data vs MSE (No. of epochs=10).

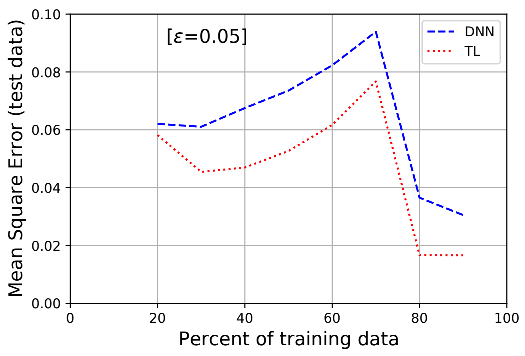

Figure E.4: Percent of training data vs MSE (No. of epochs=10).

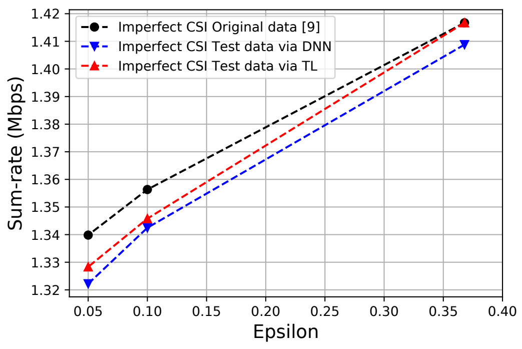

Figure E.5: Sum-rate for different values of outage probability ϵ.

Figure E.5 shows the plot of sum-rate obtained for different values of outage probability ϵ, that is, 0.05, 0.1 and 0.3679, for the case of imperfect CSI [72]. The black dashed line is the sum-rate obtained by averaging the 100,000 realizations of the original solutions calculated numerically using optimization method [72]. The blue and red dashed lines, respectively, are the sum-rates obtained by training the imperfect CSI model via DNN and TL with 50% training-testing separation of samples. It can be noticed that the sum-rate of TL is higher than the one obtained for the DNN, showing a better performance. Specifically, the TL is achieving the same sum-rate as the original for $\epsilon = 0.3679$, a condition in which the two resource allocation tasks for perfect CSI and imperfect CSI (under outage probability $\epsilon = 0.3679$) are similar. This signifies that even with a small number of training samples, TL achieves nearly the same performance as the original, and superior performance, as compared to using a DNN without TL, hence resulting in saving computational resources.

Table E.3 presents the comparison of computational performance of the proposed TL-based approach with the DNN-based approach (both implemented in Python) and with the numerical optimization methods proposed in [70, 72] (implemented in MATLAB). It can be noticed that the MATLAB implementation of both algorithms requires substantially more computational time as compared to the TL-based approach. Moreover, the case of imperfect CSI requires a lesser number of epochs to

Table E.3: Computational performance for perfect and Imperfect CSI

	Optimization methods [70, 72]	DNN		TL	
Scenario	Time (second)	Train epochs (80,000 samples)	Test time (second)	Train epochs (50,000 samples)	Test time (second)
Perfect CSI	46.0	51	0.166	-	-
Imperfect CSI	214.0	13	0.161	19	0.125

train the DNN as compared to the perfect CSI for the same training data. The TL-based approach requires a lesser number of training iterations, that is, 950,000 (19 epochs $\times$ $50,000$ samples) as compared to the imperfect CSI, which is 1040,000 (13 epochs $\times$ $80,000$ samples). As expected, the TL-based approach requires less training data (sample complexity), thus saving computational resources.

E.6 Conclusion and Future Work

In this paper, we have first designed a DNN-based algorithm for wireless resource allocation in the case of a perfect CSI model. The optimal DNN model is considered as the baseline for transfer learning in order to train an imperfect CSI model. Our results show that for the joint problem of channel assignments and power allocations over D2D communications, DNNs can approximate accurately this solver between cellular users and D2D pairs, with the knowledge of the channel-state-information (CSI). Moreover, the transferred model to the imperfect CSI scenario performs better than the DNN model without transfer learning and requires less amount of training data, reducing the sample complexity. Our results show that DNN has great potential to solve real-time wireless resource allocation problems and transfer learning can reduce the data-hungry nature of DNN, saving computational resources. TL can lead to a good performance across similar problems with a limited amount of training data.