Team Deep Neural Networks for Interference Channels

Paul de Kerret ‡, David Gesbert ‡, and Maurizio Filippone §
‡ Communication Systems Department, EURECOM
§Data Science Department, EURECOM

***Abstract*—In this paper[1], we propose to use Deep Neural Networks (DNNs) to solve so-called Team Decision (TD) problems, in which decentralized Decision Makers (DMs) aim at maximizing a common utility on the basis of locally available Channel State Information (CSI) *without any additional communication or iteration*. In the proposed configuration –coined Team DNNs (T-DNNs)–, the decision at each DM is approximated using a DNN and the weights of all DNNs are jointly trained, even though the implementation remains fundamentally decentralized. Turning to a practical application, the problem of decentralized link scheduling in Interference Channels (IC) is reformulated as a TD problem so that the T-DNNs approach can be applied. After adequate training, the scheduling obtained using the T-DNNs flexibly adapts to the decentralized CSI configuration to outperform other scheduling algorithms, thus proposing a novel efficient solution to a problem that has remained elusive for years.**

I. Introduction

A. Decentralized Coordination in Wireless Networks

Coordination between the Transmitters (TXs) in wireless network has received significant attention as a mean to improve Quality-of-Service (QoS) and spectral efficiency, through coordinated scheduling, interference reduction and alignment, joint beamforming, pilot coordination, and power control among many other possibilities. Coordination is often designed in a centralized setting whereby a computing node gathers all the necessary Channel State Information (CSI) from all TXs, computes utility maximizing decisions and forwards these decisions to the devices. In the realm of cellular networks, such a centralized implementation has been considered in the context of *Cloud Radio Access Network (RAN)* or *C-RAN* (See for example [1]) supported by high-end all-optical backhaul architectures.

Yet considering the high cost and relative lack of flexibility of C-RAN deployments there is rising interest for decentralized forms of coordination. In emerging ultra-flexible and heterogeneous deployment scenarios featuring access points mounted on buses, drones [2], or those allowing backhaul-less Device-to-Device communications, decentralized control becomes a highly desirable feature as it allows for cheaper, faster, and more flexible coordination schemes.

However, decentralized coordination between the TXs also comes with its own challenges. While the centralized implementation allows to perfectly share CSI from all TXs (so-called logically centralized), in contrast the devices in a decentralized setting must cope with their own local uncertainties regarding the global CSI in order to make a transmission decision, which calls for innovative algorithm designs.

Indeed, in the distributed CSI configuration, the TXs aim at cooperating on the basis of different locally available information, which can be recast as decentralized Team Decision (TD) problem [3]. The key challenge in TD stems from the fact that each decision maker (here, device) is limited by the local uncertainties (noise) affecting its view of the global CSI, making it difficult to predict the behavior of other devices with which it seeks to coordinate its actions. In prior works, attempts to derive noise-robust policies are reported [4], yet always relying on scenario specific policy models and heuristics, making the implementation non-generic. In [5], [6] more generic approaches are proposed. Yet, they rely on discretization thus making the approaches non-scalable.

In this work, we show how *Team* Deep Neural Networks (DNNs) under a suitable learning strategy allow for a generic approach to robust decentralized coordination. The principles of our approach are exemplified through the case of link scheduling in wireless interference channels [7].

We first give the reader a quick background on classical DNNs before moving to the newer decentralized scenario.

B. Supervised Deep Learning: A (Very) Short Overview

In supervised deep learning, we aim at *learning* a mapping f between $\boldsymbol{x}_i$ and $f(\boldsymbol{x}_i)$ from a training data set $(\boldsymbol{x}_i, f(\boldsymbol{x}_i))_{i=1}^{n}$. In this work, we model the function using *Deep Neural Network (DNN)* such that the function is restricted to be obtained from the output of a multiple-layer feed-forward DNN. A DNN consists of multiple layers where the jth layer contains n_j nodes. The output of each node is then obtained from a linear combination of the outputs of the previous layer followed by the application of a so-called *activation* function which introduces the required non-linearity. Thus, a DNN is obtained from the composition of non linear functions, where each function is a linear combination of activation functions.

Specifically, let us denote by y_i^j the output of the ith node of layer j and by Φ the activation function. The output of node i of layer j is then given by

$$y_i^j = \Phi\left(\sum_{i=1}^{n_j} \theta_i^{j-1} y_i^{j-1}\right) \tag{1}$$

[1]D. Gesbert and P. de Kerret are supported by the European Research Council under the European Union's Horizon 2020 research and innovation program (Agreement no. 670896).

where θ_i^j for all values of i and j form the parameters of the DNN that need to be *trained*. Clearly, the last layer has a number of node corresponding to the output space while the first layer corresponds to the input space. Activation functions are chosen so as to reach the desired accuracy in the training at the fastest rate. Currently, the most widely used activation function is the so-called *ReLu* function given by

$$\mathrm{ReLu}(z) = \max(z, 0). \tag{2}$$

One important advantage of the ReLu function is that its derivative is either 0 or 1 and hence easily implemented. DNNs have been known for many years but were notably difficult to train until recent breakthroughs both in terms of hardware and in terms of algorithms, which made possible computationally efficient training of DNNs [8], [9].

C. DNN Literature Overview

DNN has been applied to many different scenarios, achieving striking performances and successes [9], [10] yet mostly related to computer vision or speech processing. Decentralized implementation of reinforcement learning have recently attracted a lot of attention (See among others [11], [12]). Yet, these works rely on a markov decision process model and cannot be applied to our setting, as it will become clear through this work.

Turning to wireless communication, the application of the new deep learning tools is only at its infancy, although recently gaining a lot of momentum. DNNs have been used in some cases to reproduce (approximate) known algorithms. The advantage of this approach is that the demanding computations are then done during the training of the DNNs. Once the DNNs coefficients are obtained, the use of the DNN requires only very simple computations, thus allowing for quasi-real time processing. This approach is studied in [13] for caching and in [14] for resource allocation in interference channels.

In [15] a framework to incorporate machine learning in cognitive radio is described while its application to coding is discussed in [16], [17]. In [18], deep learning is used for detection to reduce complexity while it is used in [19] to make up for the absence of channel model while performing detection in molecular communication. In [20], learning is applied to the determination of the optimal cell-load in wireless networks by deriving and exploiting mathematical properties of the problem considered.

In [21], [22], the use of deep learning to design the physical layer is discussed and it is in particular shown how the TX, the channel, and the RX can been seen as a single DNN and trained as an autoencoder, which is a model combining encoding and decoding that are learned jointly.

D. Main Contributions

In contrast to the previous literature on deep learning for communication systems, we propose in this work the use of a novel Team DNNs (T-DNNs) with the specific goal to enable robust transmission schemes in a decentralized multi-device setting with uncertainties. To the best of our knowledge, such an application has not been considered before. In this context we bring the following contributions:

- We consider T-DNNs consisting of multiple parallel DNNs that are *centrally* trained to achieve efficient coordination with *distributed* information.
- We propose a suitable training strategy for the T-DNNs allowing for robustness with respect to an arbitrary configuration of channel feedback uncertainties existing at each device (decision maker).
- We apply the above principle to the example of link scheduling in wireless interference channels with arbitrary noisy channel feedback at each TX. We show how the scheduling obtained using the trained T-DNNs outperforms other scheduling schemes in terms of network sum throughput.

II. Team Decision Problem: A Primer

A. Team Decision Scenario

We now provide briefly a formulation of TD problems in a general multi-agent optimization context. A TD optimization problem occurs every time several Decisions Makers (DM) aim at maximizing a common utility on the basis of their own information and is hence encountered in many areas of engineering such as control, economics, and networking. We follow closely the formulation of a TD problem given in [23] and formulate the TD problems from the following parameters:

- K: The number of DMs.
- $\boldsymbol{x} \in \mathbb{C}^m$: The state of the world represented by this random variable.
- $\hat{\boldsymbol{x}}^{(j)} \in \mathbb{C}^m$: The estimate at DM j of the state of the world $\boldsymbol{x}$.
- $\boldsymbol{s}_j : \mathbb{C}^m \to \mathcal{A}_j \subset \mathbb{C}^{d_j}$: The strategy of the j-th DM. It is a function which takes value in a predefined subspace $\mathcal{A}_j$ which forms the set of the possible decisions.
- $\boldsymbol{s}_j(\hat{\boldsymbol{x}}^{(j)}) \in \mathcal{A}_j \subset \mathbb{C}^{d_j}$: The decision taken at DM j for the given realization $\hat{\boldsymbol{x}}^{(j)}$.
- $f : \mathbb{C}^m \times \Pi_{j=1}^K \mathbb{C}^{d_j} \to \mathbb{R}$: The joint objective of the K DMs.
- $p_{\boldsymbol{x},\hat{\boldsymbol{x}}^{(1)},\ldots,\hat{\boldsymbol{x}}^{(K)}}$: The joint probability distribution of the state of the world and the estimates at the K DMs. This is a common knowledge shared at each DM.

A team decision problem consists in the maximization by the K DMs of the expected joint objective on the basis of their individual information:

$$(\boldsymbol{s}_1^\star, \ldots, \boldsymbol{s}_K^\star) = \operatorname*{argmax}_{\boldsymbol{s}_1,\ldots,\boldsymbol{s}_K} \mathbb{E}\left[f(\boldsymbol{x}, \boldsymbol{s}_1(\hat{\boldsymbol{x}}^{(1)}), \ldots, \boldsymbol{s}_K(\hat{\boldsymbol{x}}^{(K)}))\right] \tag{3}$$

where the maximizing is taken over all possible strategies $\boldsymbol{s}_j, \forall j \in \{1, \ldots, K\}$ and where the difficulty resides in the fact that each strategy $\boldsymbol{s}_i$ can only depend on local input $\hat{\boldsymbol{x}}^{(j)}$ and not on observations made by other DMs. Note that problem (3) reflects the fundamental assumption that DMs are not allowed to further exchange information, neither about decisions made nor about local state observations. However, the fact that the observations $\hat{\boldsymbol{x}}^{(j)}$ can be jointly correlated, and correlated with the actual system's state $\boldsymbol{x}$, makes the model very general in the sense that some arbitrary (limited) information exchange mechanism may pre-exists, but strictly prior to the decision making stage. Furthermore, solutions of such a TD problem could then be extended to allow for iterations and exchanges between DMs.

B. Existing Approaches: Naive and Locally Robust Solutions

Before introducing in the next paragraph the robust Team DNN approach, we start by presenting what are the strategies of reference.

The first approach –called the *naive* approach– simply consists in neglecting the statistical information available. Hence, each DM considers its information as *perfect* and implicitly assumes that the other DMs share the *exact same* information. The optimization problem which is solved at DM j is then

$$\begin{aligned}&\left(\boldsymbol{s}_1^{'},\ldots,\boldsymbol{s}_{j-1}^{'},\boldsymbol{s}_j^{\text{naive}},\boldsymbol{s}_{j+1}^{'}\ldots,\boldsymbol{s}_K^{'}\right)\\&=\operatorname*{argmax}_{\boldsymbol{s}_1,\ldots,\boldsymbol{s}_K}\mathbb{E}\left[f(\boldsymbol{x},\boldsymbol{s}_1(\boldsymbol{x}),\ldots,\boldsymbol{s}_K(\boldsymbol{x}))\right].\end{aligned} \tag{4}$$

A more advanced approach, called the *Locally Robust* (LR) approach– consists in taking into account the statistical information relative to the imperfect knowledge of the channel, yet neglecting the discrepancies between the estimates at the different DMs (i.e., the decentralized nature of the CSI). The optimization problem which is solved at DM j is then

$$\begin{aligned}&\left(\boldsymbol{s}_1^{''},\ldots,\boldsymbol{s}_{j-1}^{''},\boldsymbol{s}_j^{\text{LR}},\boldsymbol{s}_{j+1}^{''}\ldots,\boldsymbol{s}_K^{''}\right)\\&=\operatorname*{argmax}_{\boldsymbol{s}_1,\ldots,\boldsymbol{s}_K}\mathbb{E}\left[f(\boldsymbol{x},\boldsymbol{s}_1(\hat{\boldsymbol{x}}^{(j)}),\ldots,\boldsymbol{s}_K(\hat{\boldsymbol{x}}^{(j)}))\right].\end{aligned} \tag{5}$$

The difference between (4) and (5) comes from considering the true state-of-the-world instead of the estimate in the objective evaluation. Note that in both (4) and (5), only $\boldsymbol{s}_j^{\text{LR}}$ is implemented in practice. The other strategies are only auxiliary optimization variables.

III. Team Deep Neural Networks

The Team Decision (TD) problem formulated in (3) is a challenging problem. Specifically, there are two main technical difficulties (i) the functional nature of the optimization variable and (ii) the decentralized structure of the information. We will now show how T-DNNs can overcome these two problems.

In the proposed T-DNNs framework, the transmission strategy at TX j is parametrized by a DNN denoted by $\boldsymbol{s}^{\boldsymbol{\theta}_j}$ and taking as input the multi-user estimate $\hat{\boldsymbol{x}}^{(j)}$ and returning as output the decision $\boldsymbol{s}^{\boldsymbol{\theta}_j}(\hat{\boldsymbol{x}}^{(j)})\in\mathcal{A}_j$. Consequently, the complete strategy space is reduced to the space that can be parametrized by the DNNs. To avoid introducing any sub-optimality, it is necessary that the DNN space is large enough such that it contains the optimal strategy $s_j^\star$ or approximates it asymptotically well. This will be ensured by choosing a DNN with enough coefficients, i.e., large/deep enough. Yet, it is also necessary that the coefficients $\boldsymbol{\theta}_j$ can be *trained* efficiently to reach their optimal values. This will be further discussed in the experiments section.

With this parametrization, the original TD problem is now approximated as

$$(\boldsymbol{\theta}_1^\star,\ldots,\boldsymbol{\theta}_K^\star)=\operatorname*{argmax}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_K}\mathbb{E}\left[f\Big(\boldsymbol{x},s_1^{\boldsymbol{\theta}_1}(\hat{\boldsymbol{x}}^{(1)}),\ldots,s_K^{\boldsymbol{\theta}_K}(\hat{\boldsymbol{x}}^{(K)})\Big)\right]. \tag{6}$$

The key aspect comes from the fact that the coefficients $\boldsymbol{\theta}_j$ can be optimized using very efficient learning methods briefly introduced in Section I-B. Indeed, the instantaneous objective function f is differentiable and can be used as objective for gradient-based optimization methods. Furthermore, as all the probability density functions are assumed to be known, the training data set, denoted by $\mathcal{S}_n^{\text{train}}$ can be obtained from n Monte-Carlo realizations of the channel and the channel estimates generated according to $p_{\boldsymbol{x},\hat{\boldsymbol{x}}^{(1)},\ldots,\hat{\boldsymbol{x}}^{(K)}}$ such that

$$\mathcal{S}_n^{\text{train}}\triangleq\left\{\left(\boldsymbol{x}_i,\hat{\boldsymbol{x}}_i^{(1)},\ldots,\hat{\boldsymbol{x}}_i^{(K)}\right)|i=1,\ldots,n\right\}. \tag{7}$$

Generating this training set can be seen as approximating the expectation in (6) using Monte-Carlo realizations to yield

$$\begin{aligned}&(\boldsymbol{\theta}_1^\star,\ldots,\boldsymbol{\theta}_K^\star)\\&\approx\operatorname*{argmax}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_K}\frac{1}{n}\sum_{i=1}^{n}\mathrm{R}\left(\boldsymbol{x}_i,p_1^{\boldsymbol{\theta}_1}(\hat{\boldsymbol{x}}_i^{(1)}),\ldots,p_K^{\boldsymbol{\theta}_K}(\hat{\boldsymbol{x}}_i^{(K)})\right)\end{aligned} \tag{8}$$

where the approximation becomes exact as the number of samples n increases to infinity.

It is important to note two elements. First, the training of the strategies/decision functions is done *jointly* at all TXs, while the *implementation* of the strategies/decision functions is decentralized at each TX. This is possible as the training depends only on the statistics which are known to all TXs. Second, this approach can be seen as a particular *reinforcement learning* approach as it does not require knowing any label, but the objective function is directly maximized using the stochastic gradient.

Formulation (8) allows for the use of deep learning tools implemented in high levels packages such as TensorFlow to train the T-DNNs. Yet, the training of DNNs is known to be sensitive to DNN parameters (number of training steps, learning rate, ...) and is the focus of ongoing works.

IV. Application to Decentralized Power Control in Interference Channel

We now turn to the practical scenario of decentralized link scheduling in interference channel with distributed CSIT. After formulating this problem as a TD problem, we will show how Team DNNs can be used to efficiently tackle this otherwise difficult problem.

A. System Setting

We consider a K-user Interference Channel (IC) consisting of K single-antenna Transmitters (TXs) and K single-antenna Receivers (RXs) with RX i being only served by TX i. The gain of the wireless channel between TX j and RX i is denoted by $G_{i,j}$ and all the gains are put together to form the *channel gain matrix* $\mathbf{G}\in\mathbb{R}^{K\times K}$ where

$$\{\mathbf{G}\}_{i,j}\triangleq G_{i,j}. \tag{9}$$

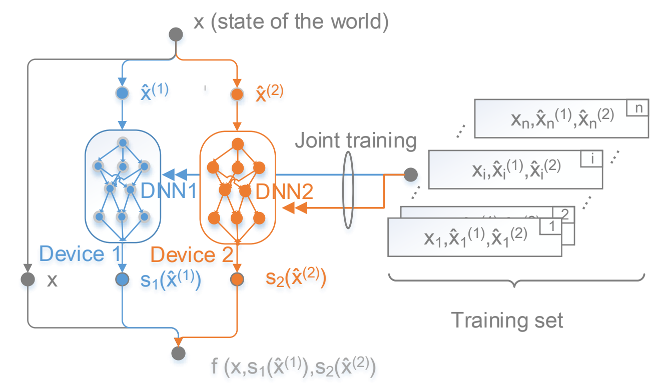


Fig. 1: Illustration of how Team-DNNs (T-DNNs) can be used to solve TD problems. Key aspects are the joint training using the common knowledge of the distribution and the decentralized application of the strategy.

The channel gain matrix can be generated according to any probability density function $p_{\mathbf{G}}$ *known to all TXs* as it is a long term information that can be estimated and shared among all TXs . We study link scheduling which means that each TX may decide between transmitting with a fixed maximum power level P or staying idle for one channel realization. This is akin to a binary power control problem whereby the power level P_j at TX j can take its values in $\{0, P\}$. Note that the proposed approach is easily extendeable to additional discrete power levels. We assume for ease of notation that all TXs have maximum power constraint P and that the RXs undergo a unit variance Gaussian noise.

We further assume that the data symbols transmitted are distributed as i.i.d. Gaussian and that each RX treats interference as noise such that the instantaneous sum rate is given by [24]

$$R(P_1,\dots,P_k)=\sum_{k=1}^{K}\log_2\left(1+\frac{G_{k,k}P_k}{1+\sum_{\ell=1,\ell\neq k}G_{k,\ell}P_\ell}\right). \tag{10}$$

Our common *welfare* objective in this work will be the expected sum rate. With perfect knowledge of the gain matrix, maximizing the expected sum rate comes down to maximizing the sum rate for each individual channel realization such that the optimal power control function $p_1^{\text{PCSI}},\dots,p_K^{\text{PCSI}}$ can be obtained from:

$$(p_1^{\text{PCSI}}(\mathbf{G}),\dots,p_K^{\text{PCSI}}(\mathbf{G}))=\operatorname*{argmax}_{(P_1,\dots,P_K)\in\{0,P\}^K}\mathrm{R}\left(\mathbf{G},P_1,\dots,P_K\right). \tag{11}$$

Binary forms of power control were shown to be rate optimal in the case of 2-user IC with perfect CSI at all TXs and near optimal with more users [7]. Importantly, perfect link scheduling at every TX requires (logically) centralized and perfect knowledge of the matrix gain $\mathbf{G}$ above.

Due to imperfect CSI feedback and limited CSI sharing links across TXs, each TX is now assumed endowed with its own imperfect estimate of the current channel state. Specifically, TX j obtains the estimate $\hat{\mathbf{G}}^{(j)}\in\mathbb{C}^{K\times K}$ of the channel gain matrix and chooses its transmission power P_j as a function of $\hat{\mathbf{G}}^{(j)}$, *without any form of information exchange with the other TXs*.

This distributed CSI model is very general as it allows for any joint distribution $p_{\mathbf{G},\hat{\mathbf{G}}^{(1)},\dots,\hat{\mathbf{G}}^{(K)}}$. The estimates at the different TXs can for example be correlated, and in the limiting case where the estimates at all TXs are exactly equal, the (logically) *centralized* CSI configuration is recovered.

B. Formulation as a Team Decision Problem

Based on the locally available channel state information $\hat{\mathbf{G}}^{(j)}$, TX j choses its binary power control P_j. Yet, the optimal instantaneous choice of P_j would normally depend on the power control decisions at the other TXs, which are unknown. Consequently, we need to introduce the power control *strategies* denoted by $p_1,\dots,p_K$ and given by

$$p_j:\begin{array}{ccc}\mathbb{R}^{K\times K} & \to & \{0,P\}\\ \hat{\mathbf{G}}^{(j)} & \mapsto & p_j(\hat{\mathbf{G}}^{(j)})\end{array} \tag{12}$$

Specifically, the TXs aim at jointly maximizing the expected sum rate, such that the TD problem formulated is given by

$$(p_1^{\star},\dots,p_K^{\star})=\operatorname*{argmax}_{(p_1,\dots,p_K)\in\mathcal{P}}\mathbb{E}\Big[\mathrm{R}\left(\mathbf{G},p_1(\hat{\mathbf{G}}^{(1)}),\dots,p_K(\hat{\mathbf{G}}^{(K)})\right)\Big] \tag{13}$$

where the expectation is carried out across all random variables, i.e., according to $p_{\mathbf{G},\hat{\mathbf{G}}^{(1)},\dots,\hat{\mathbf{G}}^{(K)}}$ and $\mathcal{P}$ is defined by

$$\mathcal{P}\triangleq\left\{(p_1,\dots,p_K)|p_j:\mathbb{R}^{K\times K}\to\{0,P\}\right\}. \tag{14}$$

Following the discussion in Section II-B, we will compare the robust T-DNNs power control function of TX j with the conventional approaches that are the *naive* power control p_j^{naive} and the *Locally Robust* power control function p_j^{LR}.

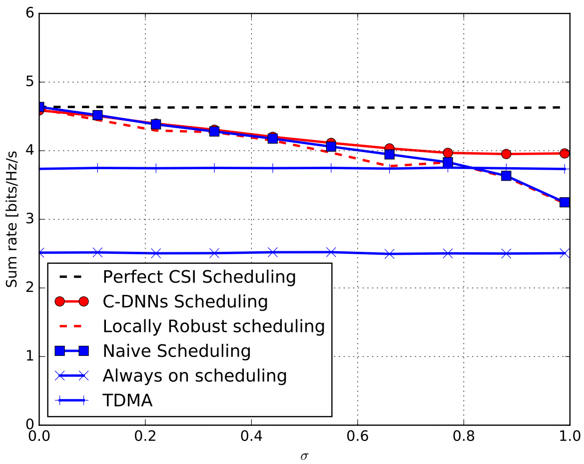
C. Team DNNs Scheduling

Now that the problem of decentralized power control with distributed CSIT has been reformulated as the TD problem (13), it is possible to apply the Team DNNs approach presented in Section III where the power control function at TX j is parametrized by a DNN denoted by $p_{\boldsymbol{\theta}_j}$ with weights $\boldsymbol{\theta}_j\in\mathbb{R}^{n_j}$ and the objective function is the sum rate.

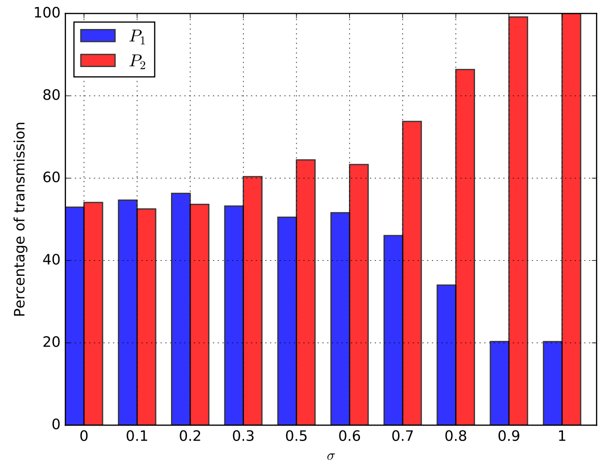
Using the right initialization is key to an efficient training [25] and we have chosen to initialize the DNN at TX j with the coefficient $\boldsymbol{\theta}_j$ that best approximates the naive link scheduling p_j^{naive}. Consequently, we start by generating a training codebook from Monte-Carlo realizations:

$$\mathcal{T}_n^{\text{naive}}\triangleq\{(\mathbf{G}_i,p_j^{\text{naive}}(\mathbf{G}_i))\}_{i=1}^{n}. \tag{15}$$

Using the training dataset $\mathcal{T}_n^{\text{naive}}$, the coefficients $\boldsymbol{\theta}_j$ can then be trained using any supervised learning classification algorithm with conventional loss function for classification (e.g., cross-entropy) [8], [9].



(a) Expected sum rate as a function of the CSI quality parameter σ.



(b) Percentage of transmission as a function of the CSI quality parameter σ. As the CSI quality at TX 1 degrades, the strategy at TX 2 becomes more deterministic with TX 1 adapting to this change.

Fig. 2: Simulation results in a 2-users IC with TX 2 having perfect CSI and TX 1 having imperfect CSI parametrized by σ.

V. Experiments

In our experiments, we use a TensorFlow implementation of a 3-layer DNN with fully connected layers comprising 30 neurons each and using the ReLu activation function defined in (2). We have used $n = 30000$ Monte-Carlo realizations with batch size of 5000 realizations. We furthermore use a drop-out probability equal to 0.5 at each node to avoid overfitting [26]. Finally, we run the Adam gradient based optimizer 10000 times with a learning rate of 0.001.

It is however important to understand that the used architecture results only from a limited trial and error approach, and could clearly be more optimized. Studying in depth what would be the optimal architecture and what would be the optimal value for the hyper-parameter is an ongoing very interesting research area but is clearly out or the scope of this paper, in which we aim only at illustrating through simulations the proposed innoative approach.

We further consider Rayleigh fading such that the channel gains are distributed as i.i.d. Chi-square random variables. To model the distributed CSI configuration, we also consider for simplicity an additive Gaussian model such that the estimate at TX j is given by

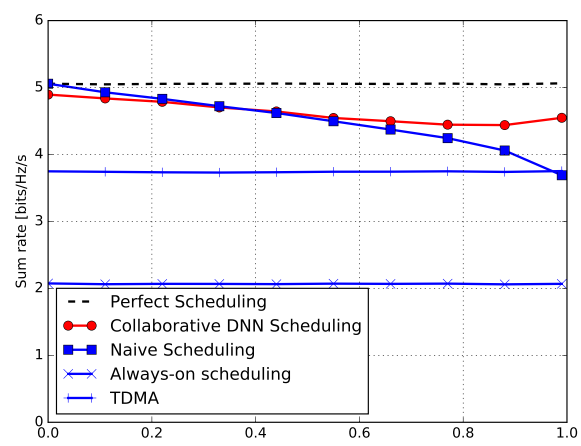
$$\hat{\mathbf{G}}^{(j)} \triangleq \bar{\mathbf{\Sigma}}^{(j)} \odot \mathbf{G} + \mathbf{\Sigma}^{(j)} \odot \mathbf{\Delta}^{(j)} \tag{16}$$

where $\odot$ is the element-wise (Hadamard) product, $\mathbf{\Delta}^{(j)}$ contains i.i.d. Chi-square random variables and $\mathbf{\Sigma}^{(j)}$ is the matrix containing the variance of the CSI noise at TX j while $\bar{\mathbf{\Sigma}}^{(j)}$ is defined such that

$$\{\bar{\mathbf{\Sigma}}^{(j)}\}_{i,k} \triangleq \sqrt{1 - \{\mathbf{\Sigma}^{(j)}\}_{i,k}^2}, \qquad \forall i,k \in \{1,\ldots,K\}. \tag{17}$$

We will compare the trained T-DNNs with the following schemes:

- *Perfect CSI scheduling*: This corresponds to the optimal link scheduling with perfect CSI instantaneously at all TXs, and is hence clearly an (a priori) loose outerbound.
- *Always-on Scheduling*: All TXs are always active.
- *TDMA*: Only one of the TX is active for all channel realizations.
- *Naive scheduling*: See Section II-B.
- *Locally Robust (LR) scheduling*: See Section II-B.



c

Fig. 3: Expected sum rate as a function of the CSI quality parameter σ in a 3-users IC with TX 1 having imperfect CSI parametrized by σ, and TX 2 and TX 3 having perfect CSI.

A. Two-User Interference Channel

Let us consider a 2-users distributed CSI configuration with the CSI noise covariance matrices given by

$$\mathbf{\Sigma}^{(1)} = \begin{bmatrix} \sigma & \sigma \\ \sigma & \sigma \end{bmatrix}, \quad \mathbf{\Sigma}^{(2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{18}$$

In Fig. 2a, the LR scheduling does not improve significantly from the naive scheduling. In contrast, T-DNNs scheduling goes as expected from naive scheduling with near perfect CSI to a solution outperforming TDMA when one TX is fully uninformed. Looking at the percentage of transmission for each TX in Fig. 2b when $\sigma = 1$, the uninformed TX transmits all the time while the informed TX adapts to this transmission using its perfect estimate to optimally balance opportunistic gain and coordination gain.

B. Three-User Interference Channel

We now turn to a 3-user IC with the following CSI noise covariance matrices:

$$\mathbf{\Sigma}^{(1)} = \sigma \mathbf{1}_{3\times 3}, \quad \mathbf{\Sigma}^{(2)} = \mathbf{\Sigma}^{(3)} = \mathbf{0}_{3\times 3}, \tag{19}$$

which means that two TXs are perfectly informed, and one TX has uniform intermediate CSI. In Fig. 3, when σ is near zero, all TXs have practically perfect CSI and T-DNNs scheduling is slightly outperformed by naive scheduling. This can be understood as the need for a larger training set, i.e., for more computing power. Furthermore, the performance of the T-DNNs scheduling seems to slightly increase with σ when σ is large, which is either due to difficulties in the training process that will be further investigated.

VI. Conclusion

In this work, we have shown how DNNs could be used collaboratively –in Team– to obtain an efficient robust solution to challenging decentralized coordination problems. When applied to decentralized link scheduling, the proposed robust solution outperforms previously known methods and is able to adapt to any distribution of channel and CSI configuration. The proposed method is very generic and could be used in many other applications and scenarios. Finally, scaling the simulations to a large number of decision makers and larger decision spaces requires developing an efficient distributed implementation of the learning algorithm.

References

[1] S. Park, O. Simeone, O. Sahin, and S. Shamai (Shitz), "Joint precoding and multivariate backhaul compression for the downlink of cloud radio access networks," *IEEE Trans. Signal Process.*, vol. 61, no. 22, pp. 5646–5658, 2013.

[2] J. Chen and D. Gesbert, "Optimal positioning of flying relays for wireless networks: A LOS map approach," in *Proc. IEEE International Conference on Communications (ICC)*, 2017.

[3] R. Radner, "Team decision problems," *The Annals of Mathematical Statistics*, 1962.

[4] Q. Li, P. de Kerret, D. Gesbert, and N. Gresset, "Robust regularized ZF in decentralized Broadcast Channel with correlated CSI noise," in *Proc. Allerton Conference on Communication, Control, and Computing (Allerton)*, 2015.

[5] P. de Kerret and D. Gesbert, "Quantized Team Precoding: A robust approach for network MIMO under general CSI uncertainties," in *Proc. IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2016.

[6] C. Zhang, V. S. Varma, S. Lasaulce, and R. Visoz, "Interference Coordination via power domain channel estimation," *IEEE Trans. Wireless Commun.*, vol. 16, no. 10, pp. 6779–6794, 2017.

[7] A. Gjendemsjo, D. Gesbert, G. E. Oien, and S. G. Kiani, "Binary power control for sum rate maximization over multiple interfering links," *IEEE Trans. on Wireless Commun.*, vol. 7, no. 8, pp. 3164–3173, 2008.

[8] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Mller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 7700. Springer*, 2012.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature International Weekly Journal of Science*, vol. 521, pp. 436–444, Apr. 2015.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature International Weekly Journal of Science*, vol. 518, pp. 529–533, Feb. 2015.

[11] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. of International Conference on Machine Learning (ICML)*, 2016.

[12] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent Actor-Critic for mixed cooperative-competitive environments," in *Proc. Neural Information Processing Systems (NIPS)*, 2017.

[13] L. Lei, L. You, G. Dai, T. Xuan, D. Yuan, and S. Chatzinotas, "A deep learning approach for optimizing content delivering in cache-enable HetNet," in *Proc. IEEE International Symposium on Wireless Communication Systems (ISWCS)*, 2017.

[14] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," 2017. [Online]. Available: https://arxiv.org/pdf/1706.01151.pdf

[15] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, "Applications of Machine Learning to Cognitive Radio Networks," *IEEE Wireless Communications*, vol. 14, no. 4, pp. 47–52, Aug. 2007.

[16] S. Cammerer, T. Gruber, J. Hoydis, and S. ten Brink, "Scaling deep learning-based decoding of polar codes via partitioning," 2017. [Online]. Available: https://arxiv.org/abs/1702.06901

[17] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *Proc. Conference on Information Sciences and Systems (CISS)*, 2017.

[18] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," 2017. [Online]. Available: https://arxiv.org/pdf/1706.01151.pdf

[19] N. Farsad and A. Goldsmith, "Detection algorithms for communication systems using deep learning," 2017. [Online]. Available: https://arxiv.org/pdf/1705.08044.pdf

[20] D. A. Awan, R. L. G. Cavalcante, and S. Stanczak, "A robust machine learning method for cell-load approximation in wireless networks," 2017. [Online]. Available: https://arxiv.org/abs/1710.09318

[21] T. OShea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cognitive Communications and Networking*, vol. PP, no. 99, 2017.

[22] S. Drner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning-based communication over the air," 2017. [Online]. Available: https://arxiv.org/abs/1707.03384

[23] G. Gnecco and M. Sanguineti, "Smooth optimal decision strategies for static team optimization problems and their approximations," in *SOFSEM 2010: Theory and Practice of Computer Science*, 2010.

[24] T. Cover and A. Thomas, *Elements of information theory*. Wiley-Interscience, Jul. 2006.

[25] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.