Power Control for Interference Management via Ensembling Deep Neural Networks

(Invited Paper)

Fei Liang*, Cong Shen*, Wei Yu† and Feng Wu*
*University of Science and Technology of China. †University of Toronto.
Email: lfbeyond@mail.ustc.edu.cn, congshen@ustc.edu.cn, weiyu@comm.utoronto.ca, fengwu@ustc.edu.cn

***Abstract*—A deep neural network (DNN) based power control method that aims at solving the non-convex optimization problem of maximizing the sum rate of a fading multi-user interference channel is proposed. Towards this end, we first present *PCNet*, which is a multi-layer fully connected neural network specifically designed for the power control problem. A key challenge in training a DNN for the power control problem is the lack of ground truth, i.e., the optimal power allocation is unknown. To address this issue, PCNet leverages a unsupervised learning strategy and directly maximizes the sum rate in the training phase. Observing that a single PCNet does not universally outperform the existing solutions, we further propose *ePCNet*, a network ensemble with multiple PCNets trained independently. Simulation results show that for the standard symmetric K-user Gaussian interference channel, the proposed methods can outperform all state-of-the-art power control solutions under various system configurations with a reduced computational complexity.**

I. Introduction

The capacity region of the multi-user interference channel is a well-known open problem in information theory [1], [2]. The interference channel is a useful model for today's wireless networks, as interference management has increasingly become the bottleneck of the overall system performance due to the broadcast nature of wireless communications and the dense deployment of base stations and mobiles, which create a heavily interfering environment.

In this paper, we focus on the power control problem of maximizing the sum rate of a multi-user interference channel, where each receiver is restricted to treating interference as noise (TIN). This problem (including some of its variations) is generally NP-hard, and has been investigated for decades. Due to its non-convex nature, state-of-the-art solutions in the literature often involve either exhaustive search (explicitly or implicitly) [3], [4] or iterative optimization of some approximate sub-problems [5]–[7]. Performance, convergence, and complexity issues hinder the practicality of these solutions. In particular, how to achieve efficient power control when the number of users is large remains an open problem.

This work addresses power control from a different perspective. Instead of directly tackling the non-convex optimization problem in an analytical fashion, we leverage the recent advances in deep learning to investigate whether a data-driven method can achieve better performance with lower complexity. In particular, the proposed approach establishes a connection between the sum-rate maximization problem and minimizing a loss function in training a deep neural network (DNN), and relies on the efficient network training and ensembling mechanism to achieve near-optimal power control.

Deep learning has had great success in computer vision, natural language processing and some other applications. Recent results also show that deep learning can be a promising tool for solving difficult communication problems, such as channel decoding [8] and channel estimation [9]. For the considered power control problem, Sun *et. al.* [10] recently proposed a neural network based method, in which the network is trained with the power control results of WMMSE [7] serving as the ground truth. This method has lower computational complexity compared to the original WMMSE, but the sum rate is also upper bounded by it. A natural question thus arises: in addition to the benefit of low complexity, can we also achieve better performance (in terms of larger sum rate) than the existing power control methods?

In this paper, we answer this question affirmatively by proposing a novel family of methods, called *ensemble Power Control Network (ePCNet)*. There are two key ingredients in ePCNet. The first is that in order to simultaneously achieve higher sum rate and lower computation complexity than existing methods, we skip the supervised learning stage, which is used in some related works [10], [11], and directly carry out the unsupervised learning for network training. This idea lifts the performance upper bound limitation of [10] and avoids the possible negative impact of the supervised learning stage of [11], which allows us to better approach the ultimate ground truth, i.e., the globally optimal power control.

The second component is *ensemble learning* which, to the best of the authors' knowledge, has not been used in the power control problem. This is particularly useful as we observe that a single PCNet may not *universally* outperform all existing methods. The proposed ePCNet, however, creates multiple independent "copies" of PCNet and trains them separately, before forming an ensemble that selects the power profile which leads to the largest sum rate. We show via simulations that combining DNN with ensemble learning results in a high-performance and low-complexity power control method that outperforms state-of-the-art methods in various system configurations.

The rest of this paper is organized as follows. The system model is introduced in Section II. In Section III, the main contributions of this paper, including the PCNet and ePCNet, are explained in detail. Simulation results are given in Section V.

The paper is concluded in Section VI.

II. System Model

We consider a general K-user single-antenna interference channel. It is assumed that all transmitter-receiver pairs share the same narrowband spectrum and are synchronized. The discrete-time baseband signal received by the i-th receiver is

$$y_i = h_{i,i}x_i + \sum_{j\in\mathcal{K}/\{i\}} h_{j,i}x_j + n_i, \tag{1}$$

where the set of transmitter-receiver pairs is $\mathcal{K} = \{1, 2, \cdots, K\}$; $\mathcal{K}/\{i\}$ denotes the set of transmitter-receiver pairs excluding the i-th one; $x_i \in \mathbb{C}$ denotes the signal transmitted by the i-th transmitter; $h_{i,i} \in \mathbb{C}$ denotes the direct-link channel for the i-th user, $h_{j,i} \in \mathbb{C}$ denotes the cross-link channel between the j-th transmitter and the i-th receiver; and $n_i \sim \mathcal{CN}(0, \sigma_i^2)$ denotes the receiver noise, which is independent across both time and users. Receiver i only intends to decode x_i. For simplicity, we assume that all receivers have the same noise power σ^2. We note that this model has been widely studied in the literature; see [3], [5], [7], [10].

A block fading channel model is assumed, i.e., the channel coefficients remain unchanged in one time slot but change independently from one time slot to another. We do not impose any limitation on the distribution of $h_{i,i}$, as our method is generic enough to handle different channel models. Random Gaussian codebooks are assumed. Encoding is independent across users and no interference cancellation is performed at each receiver. Thus, the transmitter-receiver pairs do not cooperate and multiuser interference is treated as additive noise, i.e., TIN [12]. Therefore, the effective received noise power at the i-th receiver is $\sigma_i^2 + \sum_{j\in\mathcal{K}/\{i\}} P_j\|h_{j,i}\|^2$.

The transmit power P_i for user i is chosen from set $\mathcal{P}_i \subseteq \mathbb{R}_+$. In this work, for simplicity, we consider $\mathcal{P}_i = \{P : 0 \le P \le P_{\max}\}, \forall i \in \mathcal{K}$, where $P_{\max}$ is the maximum power that transmitters can use. Note that $0 \in \mathcal{P}_i$. Thus, a user may choose not to transmit. The joint power profile of all users is denoted by a vector $\mathbf{P} = (P_1, P_2, \cdots, P_K)^T \in \mathcal{P}$, where $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2 \times \cdots \times \mathcal{P}_K$ and $(\cdot)^T$ denotes matrix transpose. For a given $\mathbf{P}$ and channel realizations $\{h_{ij}\}_{i,j\in\mathcal{K}}$, the achievable rate of the i-th receiver under Gaussian codebooks is

$$R_i(\mathbf{P}) = \log\left(1 + \frac{P_i\|h_{i,i}\|^2}{\sigma_i^2 + \sum_{j\in\mathcal{K}/\{i\}} P_j\|h_{j,i}\|^2}\right). \tag{2}$$

For each slot, the channel coefficients are fixed, and the power control algorithm outputs a power profile $\mathbf{P}$ based on the channel realization.

The objective of power control for interference management is to find the optimal power profile $\mathbf{P}$ for all users to maximize some system performance under some specific constraints. With different performance measures and different constraints, the power control problem has different formulations. In this paper, we focus on the specific one: sum rate maximization (SRM), which is formally given as

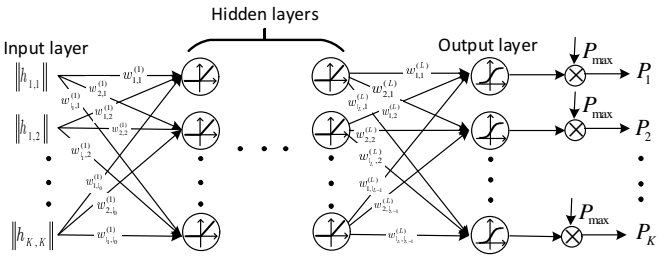


Fig. 1. Illustration of *PCNet*.

$$\begin{aligned} \underset{\mathbf{P}}{\text{maximize}} \quad & \sum_{i=1}^{K} R_i(\mathbf{P}) \\ \text{subject to} \quad & 0 \le P_i \le P_{\max}, \forall i \in \mathcal{K}. \end{aligned} \tag{3}$$

Problem (3) is the simplest formulation among various power control problems. However, it is very difficult to solve due to its non-convex nature with respect to the power profile. The optimization problem (3) is known to be NP-hard [13].

III. PCNet: Training DNN for Power Control

A. Network design

We propose PCNet by exploiting a fully connected deep neural network to address the power control problem (3). The network structure is illustrated in Fig. 1. More specifically, the network consists of one input layer of K^2 nodes, one output layer of K nodes, and $L-1$ fully connected hidden layers. These layers are indexed from 0 to L. The input layer of PCNet is formed by aligning all $\|h_{i,j}\|$ as a column vector, denoted as $\mathbf{h}$. The input vector is processed by the L fully connected layers, including $L-1$ hidden layers and one output layer.

The reason that a fully connected DNN is adopted, rather than more sophisticated networks such as convolutional neural networks (CNN) or recurrent neural networks (RNN), is that there is little structure to explore for the general problem (3). If the problem exhibits certain features a more structured neural network such as CNN may be useful. One example is in [14].

We denote the number of nodes in the k-th layer as l_k. If the k-th layer is a hidden layer, its output is calculated as follows:

$$\mathbf{c}_k = \mathsf{ReLU}\left(\mathsf{BN}\left(\mathbf{W}_k\mathbf{c}_{k-1} + \mathbf{b}_k\right)\right), \tag{4}$$

where $\mathbf{c}_{k-1}$ and $\mathbf{c}_k$ are the output vectors of the previous and current layers; their dimensions are $l_{k-1}\times 1$ and $l_k \times 1$; $\mathbf{W}_k$ is the $l_k \times l_{k-1}$ weight matrix and $\mathbf{b}_k$ is the $l_k \times 1$ bias vector; $\mathsf{BN}(\cdot)$ denotes batch normalization (BN) [15]; $\mathsf{ReLU}(\cdot)$ is the widely used Rectified Linear Unit function $(\max(x, 0))$. For the first hidden layer, we define $\mathbf{c_0} = \mathbf{h}$ and $l_0 = K^2$.

The output layer decides the transmit power of all transmitters; its calculation is different from previous layers and is given as follows:

$$\mathbf{c}_L = \mathsf{Sig}\left(\mathbf{W}_L\mathbf{c}_{L-1} + \mathbf{b}_L\right), \tag{5}$$

where $\mathsf{Sig}(x)$ denotes the standard sigmoid function:

$$\mathsf{Sig}(x) = \frac{1}{1+\exp(-x)}.$$

First, noting that the transmit power must be within the range $[0, P_{\max}]$, the sigmoid function is used as the activation function instead of ReLU to regulate the output. Second, BN is skipped in the output layer, as we have empirically observed that it would degrade the network performance. Finally, the transmit power of user i is

$$P_i = P_{\max} c_{L,i}, \tag{6}$$

where $c_{L,i}$ is the ith element of $\mathbf{c}_L$.

We note that the fully connected DNN structure can be completely captured by the number of nodes in each layer and it is concisely denoted as:

$$\{l_0, l_1, l_2, ..., l_L\}. \tag{7}$$

Remark 1: We add BN layers in the PCNet design, which was rarely considered in the existing literature for power control. We emphasize that although it is well known that employing BN layers will accelerate the training process for a general DNN, its impact goes beyond the general training and is particularly important to the considered power control problem. This is because the sigmoid layer is naturally applied to limit the output power in the range $[0, P_{max}]$. If the input value of a sigmoid node is far from 0, the gradient of the sigmoid node will vanish and the network training cannot converge. Inserting BN layers will prevent the input value of the sigmoid nodes from moving toward infinity too quickly, and hence prevent the training process from stopping earlier.

B. Training PCNet

The performance of a neural network largely depends on how it is trained. As mentioned before, in the recent work [10], the authors proposed to train the network using WMMSE as the ground truth. The loss function is defined to measure the distance between the network output and the ground truth. Obviously, the network trained with this strategy cannot outperform WMMSE, and thus the main benefit of [10] comes from the (online) computational complexity.

Ideally, were we able to obtain the globally optimal power control $\mathbf{P}^* = (P_1^*, \cdots, P_K^*)^T$ for a given channel realization, we would have designed PCNet by using the optimal solution as the ground truth. However, such approach is inapplicable considering that obtaining the global optimum efficiently is often not possible.

Noting that the sum rate is the ultimate goal of the SRM problem, we define a loss function to directly maximize the sum rate, which is

$$\mathsf{loss}_{SRM} = -\frac{1}{|\mathcal{H}|} \sum_{\mathbf{h} \in \mathcal{H}} R(\mathbf{h}, \boldsymbol{\theta}), \tag{8}$$

where $\mathcal{H}$ is a mini-batch of training samples and $|\mathcal{H}|$ denotes the number of samples in $\mathcal{H}$. Therefore, in each iteration, (8) is used for gradient descent optimization.

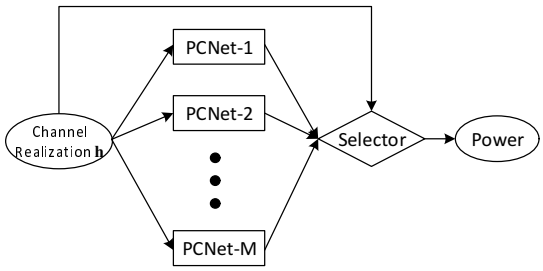


Fig. 2. ePCNet with an ensemble of M PCNets.

IV. ePCNet: Ensembling PCNets

Training PCNet with sufficiently representative data and the new loss functions defined in (8) cannot guarantee that PCNet outputs the globally optimal power profile $\mathbf{P}(\mathbf{h})^*$ for a given channel realization $\mathbf{h}$. Due to the inherent deficiency of gradient descent, the trained PCNet may fall into a local optimum. As will be corroborated in the numerical experiments, a single PCNet does not *universally* outperform existing methods.

To boost the power control performance and approach the global optimum, we incorporate the idea of *ensemble learning* and propose *ensemble PCNet*, i.e., *ePCNet*. A pictorial illustration is given in Fig. 2. Note that ensemble learning [16] is a commonly used machine learning method to achieve better performance by combining multiple *weak* local learners, each of which is powerful only for certain local use cases. Similarly, for the power control problem, we propose to build an ensemble of PCNets to achieve a better performance. Although each individual PCNet is not "powerful" enough, i.e., it cannot universally achieve better performance, a carefully formed ensemble of these weak local learners may be able to "ride the peak" of individual PCNets and output a universally better power profile.

Assume that we have M PCNets trained for a given interference channel model. For simplicity, we further assume that all PCNets have the same network structure. Operationally, this simplification also facilitates the deployment of the proposed ePCNet, since all local learners share the same computational architecture. Each of the local PCNets is trained with a different set of initial parameters, together with a set of independently generated training data. These measures are taken to enhance the *diversity* of the M local learners, which is an important principle in ensemble learning [16]. The network input $\mathbf{h}$ is first fed into all PCNets in the ensemble and the m-th PCNet outputs a power control result $\mathbf{P}_m(\mathbf{h})$. The selector collects all M outputs $\{\mathbf{P}_m(\mathbf{h}), \forall m = 1, \cdots, M\}$ as well as the channel vector $\mathbf{h}$, then computes the sum rate for each PCNet. The selector outputs the power profile with the highest sum rate, i.e.,

$$\mathbf{P}_{ePCNet}(\mathbf{h}) = \underset{\mathbf{P}_m(\mathbf{h})}{\arg\max}\, R_{PCNet}(\mathbf{h}, \mathbf{P}_m(\mathbf{h})), \tag{9}$$

where $\mathbf{P}_{ePCNet}(\mathbf{h})$ is the output power profile of ePCNet, $R_{PCNet}(\mathbf{h}, \mathbf{P}_m(\mathbf{h}))$ is the achieved sum rate of PCNet with the channel realization $\mathbf{h}$ and the power profile $\mathbf{P}_m(\mathbf{h})$.

We make the following remarks on the proposed ePCNet. First, it is well-known that training a neural network with stochastic gradient descent may often lead to local optima. Thus, combining several local weak DNNs effectively looks at multiple local optima and leads to significantly improved sum rate performance. Second, PCNet only requires a few matrix multiplications and is of low computational complexity[1]. Therefore, combining several PCNets only increases the online complexity in a linear fashion with respect to the total number of local learners. Furthermore, all networks are trained independently and training can be conducted in parallel to reduce the overall running time. Last but not the least, creating diversity by training multiple PCNets is applicable not just to DNN. Some existing methods for power control may also benefit from such *diversity*. For example, a random initialization is often needed to start an iterative algorithm. Giving such algorithms multiple random initializations may also produce multiple local optima and selecting one with the best performance would also help. However, such performance improvement is often achieved while sacrificing computational efficiency.

V. Numerical Evaluations for SRM

In this section, we intend to compare the performance and the computational complexity of various methods.

A. Schemes for comparison

1) PCNet and ePCNet: We implement the proposed PCNet and ePCNet in *TensorFlow*, following the detailed descriptions in Section III and IV. Per common practice, we use Xavier initialization [17] to initialize the network parameters and ADAM [18] to train the network. The stochastic gradient descent is used to calculate the gradient and the mini-batch size is set to 10^3. A total of 10^5 iterations are executed to train the network. The training data can be easily generated based on the channel model. As the mini-batch size is 10^3 and total iteration number is 10^5, a total of 10^8 channel samples are generated for training the network.

For SRM, it is observed that PCNet almost always outputs power profiles close to binary power control, which is also observed in another work [10]. Therefore, we round the power profiles of PCNet to binary ones. Although this will cause a small deviation from the true performance, the impact is very minor. Binary power control facilitates practical implementation as it is easy for transmitters to configure binary transmit power and hence user scheduling is implicitly considered.

2) Round-Robin Power Control (RR): The RR algorithm proposed for SRM in [5] is implemented for comparison. The basic idea of RR is to update the power of one user while keeping others fixed. This sub-problem is addressed by solving a polynomial function optimization. The algorithm stops when the following condition is satisfied,

$$\frac{|R(\mathbf{P}^{(t)}) - R(\mathbf{P}^{(t-1)})|}{R(\mathbf{P}^{(t-1)})} \leq 10^{-4}, \tag{10}$$

[1]This is also observed and numerically validated in [10].

where $R(\mathbf{P}^{(t)})$ and $R(\mathbf{P}^{(t-1)})$ denotes the sum rates in the current and last iterations respectively.

3) Iteratively Weighted MMSE (WMMSE): WMMSE is proposed to solve the sum-utility maximization problem for a MIMO interfering broadcast channel [7]. Its simplified version can be used to solve the considered SRM problem. In the literature, WMMSE is also used to generate the ground truth for the network training in [10]. Note that the same stopping condition (10) is also used for WMMSE.

4) Greedy Binary Power Control (GBPC): The greedy binary power control method in [6] is also implemented for comparison. It has been shown in [6] that GBPC can provide desirable performance that approaches the optimal binary power control in some instances.

B. Performance comparison

We currently focus on the symmetric interference channel model with i.i.d. Rayleigh fading for all channels. For the symmetric Rayleigh fading IC model, we have $h_{i,j} \sim \mathcal{CN}(0,1), \forall i,j \in \mathcal{K}$. The noise power at each receiver is set to the same σ^2. These settings are widely used in the literature [3], [5], [7], [10]. Without loss of generality, we assume $P_{\max} = 1$ and define the $\mathsf{EsN0}$ as

$$\mathsf{EsN0} = 10\log\left(\frac{P_{\max}}{\sigma^2}\right). \tag{11}$$

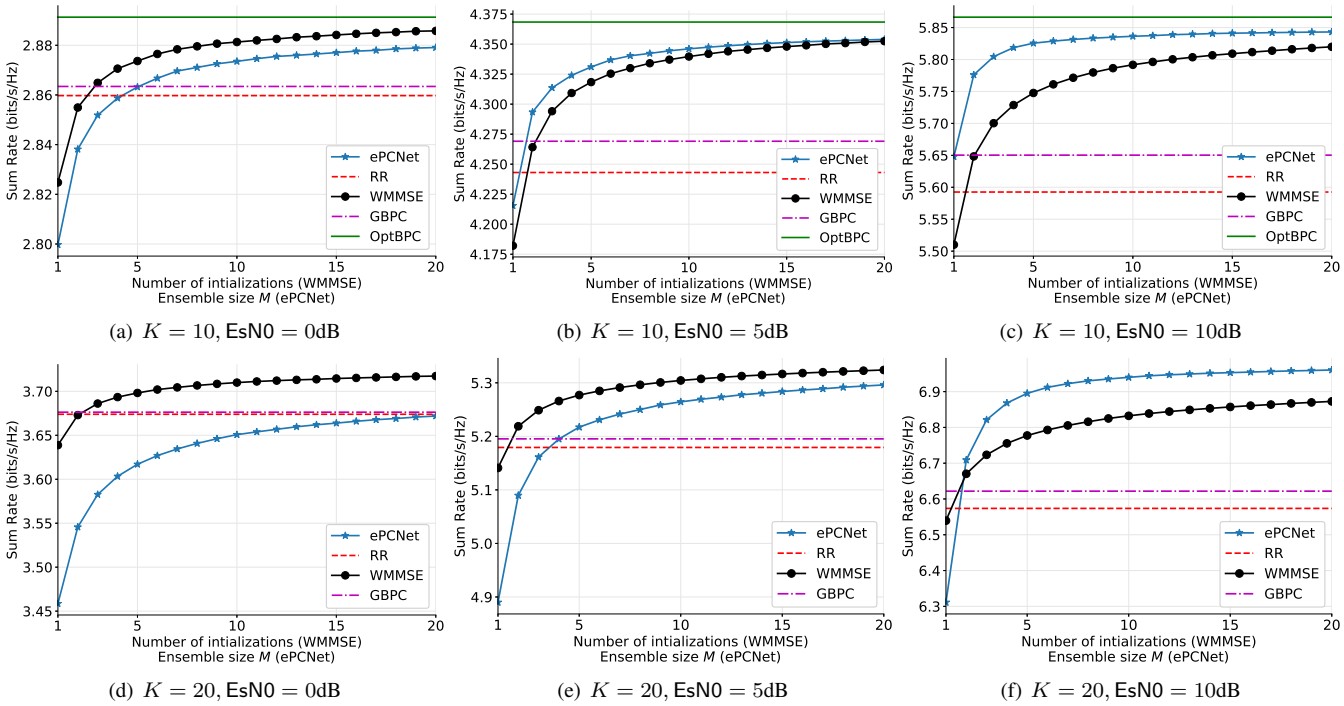
For PCNet/ePCNet, we present the results under three typical values of $\mathsf{EsN0}$, 0dB, 5dB and 10dB.

We first focus on evaluating ePCNet for the SRM problem. In this case, the considered compared algorithms include RR, WMMSE and GBPC. We use $R_{PN}(\mathbf{h})$ to denote the achievable sum rate of one PCNet or ePCNet, given channel coefficients $\mathbf{h}$. Correspondingly, $R_c(\mathbf{h})$ denotes the sum-rate of a compared scheme. We evaluate the performance of ePCNet from different perspectives.

First, we focus on comparing the achievable average sum rates of all methods, i.e., $\mathbb{E}_{\mathbf{h}}(R_{PN}(\mathbf{h}))$ and $\mathbb{E}_{\mathbf{h}}(R_c(\mathbf{h}))$. The results are shown in Fig. 3 under different system settings. Specifically, for evaluating ePCNet, we show the performance of a network ensemble of different sizes to highlight how the overall performance of ePCNet (the ensemble) scales with the number of PCNets (its local learners). The network structures for PCNet are $\{400, 400, 200, 20\}$ and $\{100, 200, 100, 10\}$ for $K = 20$ and $K = 10$, respectively.

As we mention before, WMMSE is an iterative method and it needs a random initialization to start the algorithm. If complexity is not an issue, giving WMMSE multiple random initializations would also generate multiple local optima and selecting one with the best performance would also be a strategy to achieve better performance. Therefore, as comparison, we also evaluate the performance of WMMSE with multiple random initializations.

RR is an iterative method for which multiple initializations is also possible. However, it finds a local optimum by solving a high-order equation and is of extremely high computational complexity. Thus, we do not consider multiple initializations for the RR algorithm. As for GBPC, it is a deterministic



(a) $K = 10, \mathsf{EsN0} = 0\text{dB}$ (b) $K = 10, \mathsf{EsN0} = 5\text{dB}$ (c) $K = 10, \mathsf{EsN0} = 10\text{dB}$

(d) $K = 20, \mathsf{EsN0} = 0\text{dB}$ (e) $K = 20, \mathsf{EsN0} = 5\text{dB}$ (f) $K = 20, \mathsf{EsN0} = 10\text{dB}$

Fig. 3. Sum rate comparison of different power control methods. The top row is for $K = 10$ and the bottom row is for $K = 20$. The legend *OptBPC* denotes the optimal binary power control via an exhaustive search. The performance of WMMSE is presented with multiple random initializations.

algorithm and no randomness is contained. Hence, multiple random initializations are not possible for GBPC. The performances of RR and GBPC are plotted as horizontal lines for comparison. When K is not very large, it is possible to obtain the optimal binary power control via exhaustive search. We thus also plot the performance of optimal binary power control for $K = 10$ in Fig. 3.

We first focus on comparing PCNet/ePCNet and WMMSE with multiple random initializations. A general observation is that PCNet/ePCNet can outperform WMMSE when the user number is not too large and when SNR is high. As shown in Fig. 3(b), Fig. 3(c) and Fig. 3(f), the ePCNet with ensemble size as 10 outperforms WMMSE with 10 random initializations by 0.14%, 0.77% and 1.57% in these three cases. We comment that the performance improvement of WMMSE with multiple random initializations comes with larger (online) complexity. If the computational resource is constrained and only one initialization is allowed, the performance gain of ePCNet becomes more obvious. As shown in Fig. 3(c) and Fig. 3(f), ePCNet with 10 networks outperforms WMMSE by 5.92% and 6.12% in these two cases.

We note that ePCNet does not necessarily achieve a performance gain when the user number K is large, as shown in Fig. 3(b) and Fig. 3(e). This can be explained by the fact that training a neural network of large size is not an easy task. We also note that WMMSE performs quite well when SNR is low. One possible explanation is that the background noise can smooth over those local optima of the optimization space. Thus at low SNR, WMMSE does not suffer severely from being stuck at a steep local optimum.

For the comparison between PCNet/ePCNet and other algorithms, it can be observed from Fig. 3 that GBPC achieves the best performance among the three traditional methods if only one local optima is allowed in WMMSE and RR. When further compared to PCNet and ePCNet, we have the following observations. First, a single PCNet cannot achieve universally better sum rate than existing methods. There exists a performance gap between GBPC and a single PCNet. Second, when ePCNet is used and the ensemble size increases, it can quickly outperform standard WMMSE (with single initiation), RR, GBPC in most cases except the case $K = 20, \mathsf{EsN0} = 0\text{dB}$. When $K = 20, \mathsf{EsN0} = 10\text{dB}$, ePCNet with $M = 2$ is already capable of outperforming GBPC. Even in the relatively difficult case of $K = 20, \mathsf{EsN0} = 5\text{dB}$, ePCNet with $M = 5$ has the best performance. Third, taking the three system settings shown in Fig. 3(c), Fig. 3(f), and Fig. 3(e) as examples, the sum rate gain of ePCNet with $M = 10$ over GBPC is 3.5%, 4.6% and 1.2%, respectively. Further increasing M continues to help, but the improvement becomes marginal. Furthermore, from Fig. 3(c), we find that ePCNet achieves near-optimal performance with respect to the optimal binary power control.

C. Complexity analysis

Comparing the complexity of neural networks and traditional communication algorithms is a difficult task. Simply looking at the number of floating-point operations (FLOP) is not enough for a fair conclusion, as how these operations are executed and what architecture is used also have profound impact. We note that generally PCNet has more FLOPs than WMMSE, but PCNet also has higher degree of parallelism

K, EsN0	PCNet-TF[2]	PCNet-NP[3]	WMMSE	RR	GBPC
10, 10dB	2.7	1.5	3.8	261	28
20, 10dB	3.5	8.4	5.7	1000	112
20, 5dB	3.3	7.8	5.2	1020	112

TABLE I
COMPARISON OF THE RUNNING TIME (SECONDS) OF ALL CONSIDERED METHODS.

while WMMSE is an iterative method and it has to wait for the completion of one iteration to execute the next iteration. More importantly, the implementation of neural networks has been highly optimized nowadays in libraries such as *TensorFlow*. A comprehensive comparison, especially in real-world platforms, is an important topic that is worth further investigation. We also comment that although training PCNet is time sconsuming, it only takes place in an *offline* setting.

In this subsection, we perform an approximate comparison in terms of the running time of the inference task for all methods in the same computational environment. This may not be totally accurate but it can give us at least some qualitative complexity comparison. We implement all power control methods in Python with NumPy [19] for algebra calculations. Specially, two version of PCNet are implemented in Python with TensorFlow [20] and NumPy respectively. All programs are run using the same Intel Core i7-6700 processor (3.40GHz). To handle the potential problem that different programs may have different CPU utilizations, we only enable a single CPU core for all simulations, and multi-core processing is not allowed.

In Table I, we report the running time of different schemes for 10^4 channel samples. First, by comparing PCNet-TF and PCNet-NP, we find that the running time of PCNet depends on which library is used for implementation. When $K = 10$, NumPy performs better while TensorFlow is more efficient when $K = 20$. Compared with WMMSE, PCNet-TF has less running time but PCNet-NP runs a little slower. Overall, PCNet and WMMSE are comparable in terms of running time. However, if PCNet is executed in a highly parallel platform such as GPU, its running time will be much less than WMMSE. In addition, RR and GBPC are much more computationally complex. Note that the computation time of training is not accounted for here, since training is done off-line.

VI. CONCLUSION

This paper studies a long-standing optimization problem from a new machine learning perspective. We first develop *PCNet* – a fully connected multi-layer neural network which takes the channel coefficients as input and outputs the transmit power of all transmitters. An unsupervised learning strategy is adopted to train PCNet by directly maximizing the system sum rate. An ensemble of PCNets, i.e., *ePCNet*, is proposed and shown to improve the sum-rate performance over the traditional expert-based methods. Extensive experiments have been carried out to verify the performance of ePCNet. Sum rate and complexity comparison show that ePCNet achieves better power control while consuming less computational resources. There are several directions worth further research in the future, such as different problem formulations, the network generalization to different scenarios, robustness to channel estimation errors, distributed power control and so on.

[2]PCNet-TF means PCNet is implemented with TensorFlow.

[3]PCNet-NP means PCNet is implemented with NumPy.

REFERENCES

[1] T. Han and K. Kobayashi, "A new achievable rate region for the interference channel," *IEEE Trans. Inf. Theory*, vol. 27, no. 1, pp. 49–60, Jan 1981.

[2] R. H. Etkin, D. N. C. Tse, and H. Wang, "Gaussian interference channel capacity to within one bit," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5534–5562, Dec 2008.

[3] L. Qian, Y. Zhang, and J. Huang, "MAPEL: Achieving global optimality for a non-convex wireless power control problem," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1553–1563, 2009.

[4] L. Liu, R. Zhang, and K.-C. Chua, "Achieving global optimality for weighted sum-rate maximization in the K-user Gaussian interference channel with multiple antennas," *IEEE Trans. Wireless Commun.*, vol. 11, no. 5, pp. 1933–1945, 2012.

[5] C. S. Chen, K. W. Shum, and C. W. Sung, "Round-robin power control for the weighted sum rate maximisation of wireless networks over multiple interfering links," *Transactions on Emerging Telecommunications Technologies*, vol. 22, no. 8, pp. 458–470, 2011.

[6] A. Gjendemsjø, D. Gesbert, G. E. Øien, and S. G. Kiani, "Binary power control for sum rate maximization over multiple interfering links," *IEEE Trans. Wireless Commun.*, vol. 7, no. 8, 2008.

[7] Q. Shi *et al.*, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, 2011.

[8] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 144–159, 2018.

[9] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, 2017.

[10] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438–5453, 2018.

[11] W. Lee, M. Kim, and D.-H. Cho, "Deep power control: Transmit power control scheme based on convolutional neural network," *IEEE Commun. Lett.*, vol. 22, no. 6, pp. 1276–1279, 2018.

[12] N. Naderializadeh and A. S. Avestimehr, "ITLinQ: A new approach for spectrum sharing in device-to-device communication systems," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1139–1151, 2014.

[13] M. Chiang *et al.*, "Power control in wireless cellular networks," *Foundations and Trends in Networking*, vol. 2, no. 4, pp. 381–533, 2008.

[14] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1248–1261, 2019.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. International Conference on Machine Learning*, 2015, pp. 448–456.

[16] T. G. Dietterich, "Ensemble learning," *The handbook of brain theory and neural networks*, vol. 2, pp. 110–125, 2002.

[17] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. International Conference on Learning Representations*, 2015.

[19] "http://www.numpy.org/."

[20] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning." in *Proc. Operating Systems Design and Implementation*, vol. 16, 2016, pp. 265–283.