Learning to Optimize with Unsupervised Learning: Training Deep Neural Networks for URLLC

Chengjian Sun and Chenyang Yang
School of Electronics and Information Engineering, Beihang University, Beijing, China
Email: {sunchengjian,cyyang}@buaa.edu.cn

***Abstract*—Learning the optimized solution as a function of environmental parameters by deep neural networks (DNN) is effective in solving numerical optimization in real time for time-sensitive resource allocation in wireless systems. Existing works of learning to optimize train the DNN with labels, which are generated by solving the optimization problems. The learned solution are often inaccurate and hence cannot be employed to ensure the stringent quality of service. In this paper, we propose a framework to learn the latent function with unsupervised deep learning, where the property that the optimal solution should satisfy is used as the "supervision signal" implicitly. The framework is applicable to both variable and functional optimization problems with constraints, which are respectively formulated to optimize variables and functions of concern. We take a variable optimization problem in ultra-reliable and low-latency communications as an example, which demonstrates that the ultra-high reliability can be supported by the DNN without supervision labels.**

***Index Terms*—Constrained optimization, unsupervised deep learning, ultra-reliable and low-latency communications**

I. Introduction

Resource allocation can be formulated as optimization problems with constraints, which are imposed by the available resource such as maximal transmit power, or quality-of-service (QoS) such as minimal data rate. Depending on the applications, the objective function, constraints and the variables of concern in the formulated problems may not be in the same timescale. If they are in the same timescale, say optimizing instantaneous-channel dependent power allocation among users under instantaneous rate constraint to maximize instantaneous sum rate, then the problem is variable optimization. If they are in different timescales, say optimizing instantaneous power allocation to maximize ergodic capacity under the average power constraint (the solution of such a problem is the classical water-filling power allocation), then the problem is functional optimization [1].

Variable optimization problems are widely known, which can be solved with convex optimization tools [2]. Since resource allocation depends on environmental parameters (e.g., channels), if closed-form solution can be found as a function of the parameters, then the optimized resources can adapt to environment with low complexity. If closed-form solutions cannot be found, say for non-convex problems, numerical algorithms such as interior point method have to be resorted to find the solution. Then, whenever the environmental parameters change, the optimization variables need to be numerically found again. Noticing the fact that the complexity in finding numerical solution is unacceptable for real-time processing, a novel approach was proposed in [3], which approximates the relation between the optimized solution and parameters by using neural networks. Based on the observation that significant computing efforts are wasted due to repeatedly solving a problem numerically under similar conditions, a deep learning framework was proposed in [4] to find the latent relationship between flow information and link usage by learning from past computation experience. To learn the optimal predictive resource allocation under the QoS constraint of video-on-demand service, a deep neural network (DNN) was designed in [5]. By training the DNNs offline, an approximate optimized solution can be obtained with low complexity online [3–5], say about 1% of the original numerical optimization [5]. Such an idea of "learning to optimize" is to learn an optimized solution as a function of parameters, which can be regarded as a kind of computing offloading over time by shifting the computations from online to offline.

Supervised learning was employed in [3–5], where the labels for training were generated from the optimized solutions. Hence, these methods actually make the optimization twice during training: generating labels by finding the optimization solution by numerical algorithm and training the DNN with stochastic gradient descent (SGD). The model parameters of DNN in [3–5] were trained to minimize the empirical mean square errors between the output of DNN and the labels, hence the performance of learning highly depends on the quality of labels. In [3, 4], the approximated solution can achieve around 90% performance of the numerical solution. Moreover, although the constraints in the optimization can be implicitly reflected in the labels, they are hard to satisfy by a trained-DNN that can only observe the constraints implicitly from data (i.e., labels). In [5], even with sophisticated DNN trained with

This work is supported by National Natural Science Foundation of China (NSFC) under Grant 61731002.

1,5000 labels generated from the global optimal solution, the QoS of each user can still not be satisfied. For the applications in [3–5] where the users can be served with best effort, such performance degradation is acceptable. For the applications with stringent QoS constraints such as ultra-reliable and low-latency communications (URLLC), is the DNN-based solutions still useful?

URLLC can support new application scenarios in cellular networks [6], which calls for stringent QoS requirements on the end-to-end (E2E) reliability (e.g., 10^{-5} packet loss probability) and latency (e.g., 1 ms) [7]. The latency in data transmission, channel training, queueing, and decoding have been considered in the literature of URLLC. However, the computing time for numerical optimization is never taken into account, which is not ignorable in the E2E latency.

In this paper, we propose a framework to find the relation between the solution of constrained-optimization problems and environmental parameters with unsupervised deep learning. The basic idea is to transform the variable optimization problem into functional optimization problem, where functions (and together with variables) rather than variables are optimized.

Functional optimization problems are not often formulated in wireless systems, despite that many problems such as cross-layer resource allocation can be naturally formulated as such type of problems [8]. Functional optimization problems are usually difficult to solve, which do not have closed-form solutions except for special cases such as water-filling power allocation. Hence, they have to be solved numerically, say by the finite element method that discretizing the function into multiple variables [9], which suffers from the curse of dimensionality.

To reduce the on-line computational complexity, we learn the solution of a constrained functional optimization problem by DNN, whose model parameters can be trained by minimize the Lagrange function, and hence the Karush-Kuhn-Tucker (KKT) conditions serve as the "supervision" implicitly. By using the property that the optimal solution should satisfy instead of the labels that satisfy the property, the DNN can learn the relation accurately. By converting variable optimization problems into equivalent functional optimization problems, the proposed framework can be used for both functional and variable optimization. We take a bandwidth minimization problem subject to the QoS requirement of URLLC as example, aimed to demonstrate how a variable optimization problem can be learnt by the unsupervised learning. Simulation results show that high reliability can be supported by the DNN trained without labels for supervision.

II. Learning to Optimize under Constraints without Supervision

In this section we introduce a framework to learn the solution of optimization problems with constraints without using labels. We first show how to learn the solutions of variable optimization problems as functions of environment parameters. Then, we provide a unified approach for both types of problems.

Consider a general constrained-optimization problem for a vector $\boldsymbol{x}\in\mathbb{R}^{D_x}$ consisting of D_x variables,

$$\min_{\boldsymbol{x}} \quad f\left(\boldsymbol{x};\boldsymbol{\theta}\right) \tag{1}$$

$$\text{s.t.} \quad \boldsymbol{C}\left(\boldsymbol{x};\boldsymbol{\theta}\right)\preceq\mathbf{0}, \tag{1a}$$

where $\boldsymbol{\theta}\in\mathcal{A}\subseteq\mathbb{R}^{D_\theta}$ is a vector of D_θ environmental parameters such as channel gains, $f\left(\boldsymbol{x};\boldsymbol{\theta}\right):\mathbb{R}^{D_x}\times\mathcal{A}\mapsto\mathbb{R}$ and $\boldsymbol{C}\left(\boldsymbol{x};\boldsymbol{\theta}\right):\mathbb{R}^{D_x}\times\mathcal{A}\mapsto\mathbb{R}^{D_c}$ are the objective function and D_c constraint functions of $\boldsymbol{x}$ and $\boldsymbol{\theta}$, respectively, and $\boldsymbol{y}\preceq\mathbf{0}$ means that each element in vector $\boldsymbol{y}$ is smaller than or equal to 0.

When the optimal solution of the problem in (1) does not have closed-form expression with respect to (w. r. t.) to $\boldsymbol{\theta}$, numerical solution has to be searched whenever $\boldsymbol{\theta}$ changes. This is not affordable for time-sensitive applications. To reduce the online computation time, an emerging technique is to find the relation between the optimal solution and parameters, i.e., $\boldsymbol{x}^*(\boldsymbol{\theta}):\mathcal{A}\mapsto\mathbb{R}^{D_x}$, by offline training with supervised learning [3–5]. In the sequel, we introduce a framework to find $\boldsymbol{x}^*(\boldsymbol{\theta})$ with unsupervised learning.

To embody the relation between $\boldsymbol{x}$ and $\boldsymbol{\theta}$, we optimize $\boldsymbol{x}(\boldsymbol{\theta})$ to minimize the expectation of the objective function in problem (1) over $\boldsymbol{\theta}$,

$$\min_{\boldsymbol{x}(\boldsymbol{\theta})} \quad \int_{\boldsymbol{\theta}\in\mathcal{A}} f\left(\boldsymbol{x}(\boldsymbol{\theta});\boldsymbol{\theta}\right)p(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} \tag{2}$$

$$\text{s.t.} \quad \text{(1a)},\ \forall\boldsymbol{\theta}\in\mathcal{A},$$

where $p(\boldsymbol{\theta})>0$ is the probability density function (PDF) of parameters $\boldsymbol{\theta}$, which can be arbitrary function that ensures the objective function to be integrable. In Appendix A, we prove the equivalence between the problems in (2) and (1) in the sense that they yield the same optimal solution.

When the problem in (2) is convex and the Slater's condition holds, the problem in (2) is equivalent to the following problem [2, 10][1],

$$\max_{\boldsymbol{\lambda}(\boldsymbol{\theta})}\min_{\boldsymbol{x}(\boldsymbol{\theta})}\ L\triangleq\int_{\boldsymbol{\theta}\in\mathcal{A}}\left[f(\boldsymbol{x}(\boldsymbol{\theta});\boldsymbol{\theta})+\boldsymbol{\lambda}^{\mathrm{T}}(\boldsymbol{\theta})\boldsymbol{C}(\boldsymbol{x}(\boldsymbol{\theta});\boldsymbol{\theta})\right]p(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} \tag{3}$$

$$\text{s.t. } \boldsymbol{\lambda}(\boldsymbol{\theta})\succeq\mathbf{0},\ \forall\boldsymbol{\theta}\in\mathcal{A}, \tag{3a}$$

where L is the Lagrange function, $\boldsymbol{\lambda}(\boldsymbol{\theta}):\mathcal{A}\mapsto\mathbb{R}^{D_c}$ is the vector of Lagrange multipliers, $(\cdot)^{\mathrm{T}}$ denotes transpose, and $\boldsymbol{y}\succeq\mathbf{0}$ means that each element of vector $\boldsymbol{y}$ is larger than or equal to 0.

The optimal solution of the problem in (3) should satisfy

[1]The convexity of the problem in (2) and the Slater's condition are sufficient conditions for the equivalence. For the case where these two conditions do not hold please refer to [11].

its KKT conditions, which can be derived as follows [10],

$$\frac{\delta L}{\delta \boldsymbol{x}(\boldsymbol{\theta})} = (\nabla_{\boldsymbol{x}} f + \nabla_{\boldsymbol{x}} \boldsymbol{C}\boldsymbol{\lambda})\, p(\boldsymbol{\theta}) = \mathbf{0},\ \forall \boldsymbol{\theta} \in \mathcal{A}, \tag{4}$$

$$\boldsymbol{\lambda} \star \boldsymbol{C} = \mathbf{0},\ \forall \boldsymbol{\theta} \in \mathcal{A}, \tag{5}$$

$$\text{(1a)},\ \text{(3a)},\ \forall \boldsymbol{\theta} \in \mathcal{A},$$

where $\frac{\delta L}{\delta \boldsymbol{x}(\boldsymbol{\theta})}$ is the variation of L w. r. t. function $\boldsymbol{x}(\boldsymbol{\theta})$ [1], and $\star$ denotes element wise multiplication.

The optimal solution of the problem in (2) is hard to find from the KKT conditions, because it is the functions $\boldsymbol{x}(\boldsymbol{\theta})$ and $\boldsymbol{\lambda}(\boldsymbol{\theta})$ that need to be optimized, which can be interpreted as the vectors with infinite elements.

Thanks to the universal approximation theorem [12], we can approximate $\boldsymbol{x}(\boldsymbol{\theta})$ and $\boldsymbol{\lambda}(\boldsymbol{\theta})$ with neural networks $\mathcal{N}_x(\boldsymbol{\theta};\boldsymbol{\omega}_x)$ and $\mathcal{N}_\lambda(\boldsymbol{\theta};\boldsymbol{\omega}_\lambda)$, respectively, where $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_\lambda$ are finite network parameters. Then, by replacing $\boldsymbol{x}(\boldsymbol{\theta})$ and $\boldsymbol{\lambda}(\boldsymbol{\theta})$ with $\hat{\boldsymbol{x}} \triangleq \mathcal{N}_x(\boldsymbol{\theta};\boldsymbol{\omega}_x)$ and $\hat{\boldsymbol{\lambda}} \triangleq \mathcal{N}_\lambda(\boldsymbol{\theta};\boldsymbol{\omega}_\lambda)$, the problem in (3) can be re-written as,

$$\begin{aligned} \max_{\boldsymbol{\omega}_\lambda} \min_{\boldsymbol{\omega}_x} \quad & \hat{L} \triangleq \int_{\boldsymbol{\theta}\in\mathcal{A}} \left(\hat{f} + \hat{\boldsymbol{\lambda}}^{\mathrm{T}} \hat{\boldsymbol{C}}\right) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \\ \text{s.t.} \quad & \hat{\boldsymbol{\lambda}} \succeq \mathbf{0},\ \forall \boldsymbol{\theta} \in \mathcal{A}, \end{aligned} \tag{6}$$

where $\hat{f} \triangleq f(\hat{\boldsymbol{x}};\boldsymbol{\theta})$ and $\hat{\boldsymbol{C}} \triangleq \boldsymbol{C}(\hat{\boldsymbol{x}};\boldsymbol{\theta})$. The KKT conditions of the problem in (6) can be derived as,

$$\nabla_{\boldsymbol{\omega}_x} \hat{L} = \int_{\boldsymbol{\theta}\in\mathcal{A}} \nabla_{\boldsymbol{\omega}_x} \hat{\boldsymbol{x}} \left(\nabla_{\boldsymbol{x}} \hat{f} + \nabla_{\boldsymbol{x}} \hat{\boldsymbol{C}} \hat{\boldsymbol{\lambda}}\right) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} = \mathbf{0}, \tag{7}$$

$$\nabla_{\boldsymbol{\omega}_\lambda} \hat{L} = \int_{\boldsymbol{\theta}\in\mathcal{A}} \nabla_{\boldsymbol{\omega}_\lambda} \hat{\boldsymbol{\lambda}} \hat{\boldsymbol{C}} p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} = \mathbf{0}, \tag{8}$$

$$\hat{\boldsymbol{C}} \preceq \mathbf{0},\ \hat{\boldsymbol{\lambda}} \succeq \mathbf{0},\ \forall \boldsymbol{\theta} \in \mathcal{A},$$

where the gradients $\nabla_{\boldsymbol{\omega}_x} \hat{\boldsymbol{x}}$ and $\nabla_{\boldsymbol{\omega}_\lambda} \hat{\boldsymbol{\lambda}}$ can be computed via back propagation.

Remark 1. To learn the optimal neural networks, we can take $\hat{L}$ as the loss function, and train $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_\lambda$ with gradient based methods such as SGD method. Since no label is used for supervision, the proposed learning method is unsupervised, where the KKT conditions implicitly serve as the "supervised signal".

Remark 2. Comparing the KKT conditions of the problem in (3) and the problem in (6), we can find that the conditions in (4) and (5) are tighter than the conditions in (7) and (8). This is because the former ones should be satisfied for each value of $\boldsymbol{\theta}$, while the later ones only need to be satisfied on average over $\boldsymbol{\theta}$. Because the numbers of equations in (7) and (8) are respectively equal to the numbers of elements in $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_\lambda$, increasing the scale of $\mathcal{N}_x(\boldsymbol{\theta};\boldsymbol{\omega}_x)$ and $\mathcal{N}_\lambda(\boldsymbol{\theta};\boldsymbol{\omega}_\lambda)$ can make the conditions in (7) and (8) tighter and closer to the original KKT conditions in (4) and (5). When the dimension of $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_\lambda$ is sufficient large, the two functions $\boldsymbol{x}(\boldsymbol{\theta})$ and $\boldsymbol{\lambda}(\boldsymbol{\theta})$ can be approximated by DNN accurately.

Remark 3. For the constrained-optimization problem for functions, we can still use the unsupervised learning framework to find the relation between the optimal solution and the parameters. The only difference lies in the inputs of the neural networks, which not only consist of the parameters but also the input variables of the functions to be optimized.

III. An Example Problem of URLLC

In this section, we illustrate how to learn the constrained variable optimization problem with a QoS-constrained bandwidth minimization problem for URLLC.

A. System Models

Consider an uplink (UL) orthogonal frequency division multiple access system, where a base station (BS) with N_{t} antennas serves K single-antenna users. The maximal transmit power of each user is $P_{\max}$. The bandwidth allocated to the kth user is W_k. Since the packet size u in URLLC is typically small (e.g., 20 bytes [6]), the bandwidth required for transmitting each packet is less than the channel coherence bandwidth. Therefore, the channel is flat fading.

Time is discretized into frames, each with duration T_{f}. In each frame, multiple packets are generated at the kth user and will be sent to the BS. The duration for UL data transmission in one frame is τ and the duration for channel training is T_{f}–τ. Since the E2E delay requirement in URLLC is typically shorter than the channel coherence time, the channel is quasi-static and time diversity cannot be exploited. To guarantee the transmission reliability within the delay bound, we consider frequency hopping, where each user is assigned with different subchannels in adjacent frames. When the frequency interval between adjacent subchannels is larger than the coherence bandwidth, the small scale channel gains of a user among frames are mutually independent.

1) Achievable Rate in Finite Blocklength Regime: In URLLC, the blocklength of channel coding is short due to the short transmission duration, and hence the impact of decoding errors on reliability cannot be ignored. Shannon's capacity formula cannot be employed to characterize the probability of decoding errors [13]. The achievable rate in finite blocklength regime is required. In quasi-static flat fading channels, when the instantaneous channel gain is available at the transmitter and receiver, the achievable rate of the kth user (in packets/frame) can be accurately approximated by [14],

$$s_k \approx \frac{\tau W_k}{u \ln 2} \left[\ln\left(1 + \frac{\alpha_k g_k P_{\max}}{N_0 W_k}\right) - \sqrt{\frac{V_k}{\tau W_k}} Q_{\mathrm{G}}^{-1}(\varepsilon_k^{\mathrm{c}})\right], \tag{9}$$

where $\varepsilon_k^{\mathrm{c}}$ is the decoding error probability of the kth user, α_k and g_k are the large-scale channel gain and small-scale channel gain of the kth user, respectively, N_0 is the single-side noise spectral density, $Q_{\mathrm{G}}^{-1}(x)$ is the inverse of

the Gaussian Q-function, and $V_k = 1-1/(1+\frac{\alpha_k g_k P_{\max}}{N_0 W_k})^2$ [14].

Although the achievable rate is in closed-form, it is still too complicated to obtain graceful results. As shown in [13], if the signal-to-noise ratio (SNR) $\frac{\alpha_k g_k P_{\max}}{N_0 W_k} \geq 5$ dB, $V_k \approx 1$ is accurate. Since high SNR is required to ensure ultra-high reliability and ultra-low latency, such approximation is reasonable. Even when the SNR is not high, we can obtain a lower bound of the achievable rate by substituting $V_k \approx 1$ into s_k. Then, when the required ε^{c} is satisfied with the lower bound, it can also be satisfied with the achievable rate in (9).

2) Quality-of-Service: When the service rate is random due to the channel fading, the packets may not be served immediately and will accumulate into a queue in the buffer at the user. The QoS requirements of URLLC can be characterized by the delay bound $D_{\max}$ and the overall packet loss probability $\varepsilon_{\max}$. The downlink transmission delay is subtracted from the E2E delay in this paper, since it has been studied in [8]. Thus, herein $D_{\max}$ is the UL delay, which consists of the queueing delay (denoted as D_k^{q} for the kth user), transmission delay D^{t} and decoding delay D^{c}. All these delay components are measured in frames. D^{t} and D^{c} are constant values [15]. Due to the random packet arrival, D_k^{q} is random. To ensure the delay requirement, D_k^{q} should be bounded by $D_{\max}^{\mathrm{q}} \triangleq D_{\max} - D^{\mathrm{t}} - D^{\mathrm{c}}$. If the queueing delay of a packet exceeds $D_{\max}^{q}$, the packet will be useless.

Denote $\varepsilon_k^{\mathrm{q}} \triangleq \Pr\{D_k^{\mathrm{q}} > D_{\max}^{\mathrm{q}}\}$ as the queueing delay violation probability. Then, the overall reliability requirement can be characterized by

$$1-(1-\varepsilon_k^{\mathrm{c}})(1-\varepsilon_k^{\mathrm{q}}) \approx \varepsilon_k^{\mathrm{c}} + \varepsilon_k^{\mathrm{q}} \leq \varepsilon_{\max}. \tag{10}$$

This approximation is very accurate, because the values of $\varepsilon_k^{\mathrm{c}}$ and $\varepsilon_k^{\mathrm{q}}$ are very small in URLLC.

B. Minimizing the Bandwidth to Ensure the QoS

In what follows, we show how to find the minimal bandwidth required by each user to support the QoS requirement in URLLC as a function of the large-scale channel gain. We first show how to find the optimal solution for each given value of the large-scale channel gain, and then show how to learn the optimal solutions for arbitrary values of the large-scale channel gains in an unsupervised manner.

1) Problem Formulation: Since the achievable rate in (9) depends on the small-scale channel gain, the packet service rate of each user is random. In this case, effective capacity can be applied to analyze the queueing delay [16],[2] with which an upper bound of the queueing delay violation probability of the kth user can be expressed as

$$\varepsilon_k^{\mathrm{q}} < e^{-\theta_k C_k^{\mathrm{E}} D_{\max}^{\mathrm{q}}}, \tag{11}$$

[2]We have validated that effective capacity can be applied in URLLC, but do not show the results due to the space limitation.

where C_k^{E} is the effective capacity of the service process of the kth user, and θ_k is the QoS exponent. Since the small-scale channel gains of a user are independent among frames owing to frequency hopping, the effective capacity of the kth user can be expressed as [17]

$$C_k^{\mathrm{E}} = -\frac{1}{\theta_k} \ln \mathbb{E}_{g_k} \left\{ e^{-\theta_k s_k} \right\} \text{ (packets/frame)}, \tag{12}$$

where the expectation is taken over small-scale channel gains.

If the upper bound in (11) is satisfied, then the queueing delay requirement $(D_{\max}^{\mathrm{q}}, \varepsilon_k^{\mathrm{q}})$ can be satisfied. Furthermore, both the delay requirement and the overall reliability requirement in (10) (i.e., the QoS requirement) can be satisfied if

$$\varepsilon_k^{\mathrm{c}} + e^{-\theta_k C_k^{\mathrm{E}} D_{\max}^{\mathrm{q}}} = \varepsilon_{\max}. \tag{13}$$

As shown in [8], the optimal values of the packet loss probabilities are in the same order of magnitude. Therefore, we set $\varepsilon_k^{\mathrm{c}} = e^{-\theta_k C_k^{\mathrm{E}} D_{\max}^{\mathrm{q}}} = \varepsilon_{\max}/2$ for simplicity.

To satisfy the upper bound in (11), the effective capacity should not be lower than the packet generation rate, i.e., $C_k^{\mathrm{E}} \geq m_k$, where m_k is the number of packets generated at the kth user in each frame. Noticing that if the upper bound is loose, then more bandwidth is required to ensure the QoS conservatively. Hence, we optimize the QoS exponent together with the bandwidth allocated to each user to minimize the total bandwidth, which yields a tighter upper bound. For given large scale channel gains, the optimal bandwidth allocation problem that minimizes the total bandwidth required to ensure the QoS of every user can be formulated as,

$$\min_{W_k, \theta_k} \quad \sum_{k=1}^{K} W_k \tag{14}$$

$$\text{s.t.} \quad -\frac{1}{\theta_k} \ln \mathbb{E}_{g_k} \left\{ e^{-\theta_k s_k} \right\} \geq m_k, \tag{14a}$$

$$s_k = \frac{\tau W_k}{u \ln 2} \left[\ln\left(1+\frac{\alpha_k g_k P_{\max}}{N_0 W_k}\right) - \frac{Q_{\mathrm{G}}^{-1}(\varepsilon_{\max}/2)}{\sqrt{\tau W_k}} \right], \tag{14b}$$

$$e^{-\theta_k C_k^{\mathrm{E}} D_{\max}^{\mathrm{q}}} = \varepsilon_{\max}/2, \tag{14c}$$

$$W_k > 0,$$

where (14a) is the queueing delay requirement, (14b) is the achievable packet rate in (9) to support the decoding reliability requirement $\varepsilon_k^{\mathrm{c}}$, and (14c) is from (13) to ensure the overall reliability requirement.

Since constraints (14a), (14b) and (14c) of a user do not rely on the bandwidth allocated to and the QoS exponents of other users, problem (14) can be equivalently decomposed into multiple problems each finding the minimum bandwidth required by each user. In what follows, we only consider the single user scenario of problem (14) and omit the subscript k for notational simplicity. Then, the relation needs to learn is $W^*(\alpha)$, i.e., the parameter in (1) is α.

2) Finding the Optimal Solution Given α: To provide a baseline for the unsupervised learning method, in what follows we show how to find the optimal solution with

given values of the large-scale channel gain. Since less bandwidth is required if the queueing delay requirement is looser, the optimal solution of the problem in (14) should be obtained when the equality in (14a) holds. Then, from the equalities in (14a) and (14c), the optimal QoS exponent can be solved as

$$\theta^* = -\frac{\ln\left(\varepsilon_{\max}/2\right)}{mD^{\mathrm{q}}_{\max}}, \tag{15}$$

and the optimal bandwidth can be found by finding the solution of the equalities in (14a) and (14b).

If effective capacity can be derived as an close-form expression, say for large scale antenna systems [18], then numerical solution can be found for the problem in (14). For general wireless systems where the effective capacity is not with the closed form, the constraint in (14a) does not have closed-form expression. For such kind of problems, we can resort to stochastic optimization methods such as SGD for minimization problems and stochastic gradient ascent (SGA) for maximization problems. To obtain an unbiased gradient estimation with stochastic gradient methods, the expectations in the objective function and constraints of a problem should not be in nonlinear forms. Thus, we transform (14a) into an equivalent form that is linear to the expectation. Further substituting θ^*, the QoS constraint becomes

$$\mathbb{E}_{g_k}\left\{e^{-\theta^* s}\right\} - e^{-\theta^* m} \leq 0. \tag{16}$$

Due to the expectation in (16) and the complex expression of the achievable rate in (14b), the monotonicity of the function in the left hand side of (16) is hard to analyze. In concept, the achievable rate should increase with the bandwidth. However, this may not be true when the small-scale channel gain is very small (lower than -10 dB) due to the approximation $V \approx 1$. Fortunately, since very small values of the small-scale channel gain rarely occur (e.g., $\Pr\{g < 0.1\} < 10^{-12}$ when $N_{\mathrm{t}} \geq 8$ for Rayleigh fading channels), the impact can be ignored after taking the expectation. Therefore, it is reasonable to assume that the left-hand side of (16) decreases with W. Then, the optimal bandwidth allocation can be found with stochastic optimization through the following iterations,

$$W^{(t+1)} = \left[W^{(t)} + \phi(t)\left(e^{-\theta^* s^{(t)}} - e^{-\theta^* B^{\mathrm{E}}}\right)\right]^+, \tag{17}$$

where $[x]^+ \triangleq \max\{x, 0\}$ ensures the results to be positive, $\phi(t) > 0$ is the step size, and $s^{(t)}$ is the achievable rate computed from the realization of g in the tth iteration according to (14b). With the aforementioned assumption (which is true as we validated via simulations) and $\phi(t) \sim \mathcal{O}\left(\frac{1}{t}\right)$, $\{W^{(t)}\}$ converges to the unique optimal bandwidth [19].

3) Learning $W^*(\alpha)$ without Supervision: In the sequel we show how to employ unsupervised learning to find the approximated relation between the optimal bandwidth allocated to a user and the large-scale channel gain of the user. The QoS exponent is not learned since it has analytical solution as in (15). We first transform the bandwidth optimization problem to the following functional optimization problem,

$$\begin{aligned} \min_{W(\alpha)} \quad & \mathbb{E}_\alpha\left\{W(\alpha)\right\} \\ \text{s.t.} \quad & \text{(14b)},\ \text{(16)},\ W(\alpha) > 0. \end{aligned} \tag{18}$$

Instead of solving the problem in (18), we resort to solving the following problem,

$$\begin{aligned} \max_{\lambda(\alpha)} \min_{W(\alpha)} \ & L \triangleq \mathbb{E}_\alpha\left\{W(\alpha) + \lambda(\alpha)\left(\mathbb{E}_{\boldsymbol{g}}\left\{e^{-\theta^* s}\right\} - e^{-\theta^* m}\right)\right\} \\ & \text{s.t. (14b)},\ W(\alpha) > 0,\ \lambda(\alpha) > 0, \end{aligned}$$

where $\lambda(\alpha)$ is the Lagrange multiplier.

Then, we approximate $W(\alpha)$ and $\lambda(\alpha)$ with fully connected neural networks $\hat{W} \triangleq \mathcal{N}_W\left(\alpha; \boldsymbol{\omega}_W\right)$ and $\hat{\lambda} \triangleq \mathcal{N}_\lambda\left(\alpha; \boldsymbol{\omega}_\lambda\right)$, respectively. By applying `Softplus` in the output layers in both neural networks, $\hat{W}$ and $\hat{\lambda}$ are automatically positive. Taking $\hat{L}$ as the loss function, which is obtained by substituting $W(\alpha) \approx \hat{W}$ and $\lambda(\alpha) \approx \hat{\lambda}$ into L, we can train $\boldsymbol{\omega}_W$ and $\boldsymbol{\omega}_\lambda$ respectively with SGD and SGA,

$$\begin{aligned} \boldsymbol{\omega}_W^{(t+1)} &= \boldsymbol{\omega}_W^{(t)} - \phi(t)\nabla_{\boldsymbol{\omega}_W}\hat{L}^{(t)} \\ &= \boldsymbol{\omega}_W^{(t)} - \phi(t)\nabla_{\boldsymbol{\omega}_W}\mathcal{N}_W\left(\alpha^{(t,n)}; \boldsymbol{\omega}_W^{(t)}\right)\frac{\mathrm{d}\hat{L}^{(t)}}{\mathrm{d}W}, \end{aligned} \tag{19}$$

$$\begin{aligned} \boldsymbol{\omega}_\lambda^{(t+1)} &= \boldsymbol{\omega}_\lambda^{(t)} + \phi(t)\nabla_{\boldsymbol{\omega}_\lambda}\hat{L}^{(t)} \\ &= \boldsymbol{\omega}_\lambda^{(t)} + \phi(t)\nabla_{\boldsymbol{\omega}_\lambda}\mathcal{N}_\lambda\left(\alpha^{(t,n)}; \boldsymbol{\omega}_W^{(t)}\right)\frac{\mathrm{d}\hat{L}^{(t)}}{\mathrm{d}\lambda}, \end{aligned} \tag{20}$$

where $\hat{L}^{(t)} \triangleq \frac{1}{N_{\mathrm{b}}}\sum_{n=1}^{N_{\mathrm{b}}}\left[\hat{W} + \hat{\lambda}\left(e^{-\theta\hat{s}^{(t,n)}} - e^{-\theta m}\right)\right]$ is the realization of $\hat{L}$ in the tth iteration, N_{b} is the number of realizations in each batch, $\alpha^{(t,n)}$ and $\hat{s}^{(t,n)}$ are the nth realizations of the large-scale channel gain and the achievable rate in the tth iteration, respectively. The gradient matrixes of the neural networks w. r. t. the parameters $\nabla_{\boldsymbol{\omega}_W}\mathcal{N}_W\left(\alpha^{(t,n)}; \boldsymbol{\omega}_W^{(t)}\right)$ and $\nabla_{\boldsymbol{\omega}_\lambda}\mathcal{N}_\lambda\left(\alpha^{(t,n)}; \boldsymbol{\omega}_\lambda^{(t)}\right)$ can be computed by backward propagation, $\frac{\mathrm{d}\hat{L}^{(t)}}{\mathrm{d}W} = 1 - \frac{1}{N_{\mathrm{b}}}\sum_{n=1}^{N_{\mathrm{b}}}\hat{\lambda}^{(t)}\theta\frac{\partial\hat{s}^{(t,n)}}{\partial\hat{W}}e^{-\theta\hat{s}^{(t,n)}}$ and $\frac{\mathrm{d}\hat{L}^{(t)}}{\mathrm{d}\lambda} = \frac{1}{N_{\mathrm{b}}}\sum_{n=1}^{N_{\mathrm{b}}}\left(e^{-\theta^*\hat{s}^{(t,n)}} - e^{-\theta^* m}\right)$.

Remark 4. From the iteration of the parameters of the neural network for Lagrange multiplier in (20), we can find that if the iteration converges then the QoS constraint in (16) will be satisfied. Extensive simulation results show that the iteration always converges for this example problem, despite that it is hard to prove the convergency.

IV. Simulation Results

In this section, we first show the tradeoff between the minimal bandwidth required to ensure the QoS. Then, we show the performance of the approximated optimal solution obtained with the unsupervised learning.

Without lose of generality, we consider a single user case in a single cell with radius of 250 m and path loss model $10\lg(\alpha)=35.3+37.6\lg(d)$. The simulation setup and fine-tuned hyper-parameters for the neural network are listed in Table I, unless otherwise specified. We use `Softplus` in the output layers in all DNNs, and use `TanH` in the hidden layers as an example activation function.

TABLE I
SIMULATION PARAMETERS AND HYPER-PARAMETERS

Overall packet loss probability $\varepsilon_{\max}$	10^{-5}
UL delay bound $D_{\max}$	10 frames (1 ms)
Duration of each frame T_{f}	0.1 ms
Duration of UL transmission τ	0.05 ms
Transmission delay D^{t}	1 frame [15]
Decoding delay D^{c}	1 frame [15]
Maximal transmit power of BS $P_{\max}$	23 dBm
Number of antennas N_{t}	8
Single-sided noise spectral density N_0	-173 dBm/Hz
Packet size u	20 bytes (160 bits) [6]
Packets generated in each frame m	1 packet
Learning rate $\phi(t)$	0.1
Number of hidden layers	4
Number of neurons in each layer	8
Batch size N_{b}	100

TABLE II
MINIMAL BANDWIDTH REQUIRED BY DIFFERENT RELIABILITY REQUIREMENTS (MHZ), W^*.

Required $\varepsilon_{\max}$	10^{-4}	10^{-5}	10^{-6}	10^{-7}
$d=250$ m	0.512	0.527	0.542	0.556
$d=50$ m	0.204	0.208	0.211	0.213

Table II shows the minimal bandwidth required to ensure the required QoS, where several overall packet loss probability requirements are considered. The results are obtained by finding the optimal solution given α (i.e., the distance d). From the results we can find that the minimal bandwidth is insensitive to the required reliability. For the user located at cell-centre and the user at cell-edge, only around 3% of additional bandwidth is required to enhance the reliability by one order of magnitude.

To evaluate the results of $W^*(\alpha)$ learned without supervision, $\mathcal{N}_W(\alpha;\boldsymbol{\omega}_W)$ and $\mathcal{N}_\lambda(\alpha;\boldsymbol{\omega}_\lambda)$ are trained according to the iterations in (19) and (20). In each iteration, N_{b} realizations are generated, where the small-scale channel gain are randomly generated from Rayleigh distribution, and the large-scale channel gains are computed from the path loss model with user-BS distance d uniformly distributed in $50\sim250$ m.

Fig. IV shows how the approximation accuracy, defined as $\sigma\triangleq\sqrt{\mathbb{E}_\alpha\{(\hat{W}-W^*)^2\}/\mathbb{E}_\alpha\{(W^*)^2\}}$, changes with iterations, where W^* is obtained by finding the optimal solution given α with (17). Each curve is obtained by averaging over 100 times of training. From the results we can find that the learnt solution can well approximate the optimal solution, and the convergence speed is insensitive to the reliability requirement. Specifically, $\sigma<1\%$ after around 6 000 iterations for a wide range of the reliability requirement.

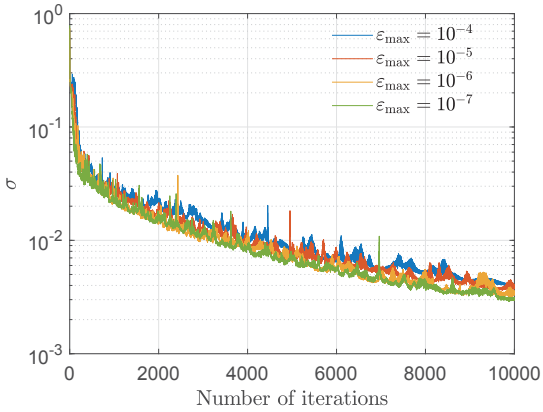

Fig. 1. Convergence of the proposed unsupervised learning.

Due to the stochastic optimization, the learning based solution is unable to satisfy the QoS requirement with probability 1. In the sequel, we evaluate the availability, which is the probability that the QoS requirement in (16) can be satisfied. Since both the minimal bandwidth and the convergence rate are insensitive to the reliability requirement, we have a chance to improve the availability with minor sacrifice of bandwidth and higher training complexity. In particular, we reduce the overall reliability requirement during training, denoted as ε_{D}, from the required overall reliability, i.e., $\varepsilon_{\mathrm{D}}<\varepsilon_{\max}$. With such a conservative design, even if the overall packet loss probability achieved by the learnt solution violates ε_{D}, it is less likely to violate the original reliability requirement $\varepsilon_{\max}$. To evaluate the bandwidth loss, we define the 1st percentile bandwidth loss as $\tilde{W}\triangleq\inf\{x|\Pr\{\hat{W}-W^*\}\leq1\%\}$, which means that the bandwidth loss is lower than $\tilde{W}$ in 99% cases.

TABLE III
AVAILABILITY AND BANDWIDTH LOSS IN ENSURING $\varepsilon_{\max}=10^{-5}$.

ε_{D}	10^{-5}	10^{-6}	10^{-7}
Availability	46.7%	98.9%	99.96%
$\tilde{W}$	1.3%	3.3%	5.7%

Table III shows the availability and the 1st percentile bandwidth loss for different values of ε_{D}. The results are obtained with 1 000 times of training, each with 10 000 iterations. The bandwidth loss and the availability of each training result are evaluated over 1 000 randomly generated large-scale channel gains. We can see that setting $\varepsilon_{\mathrm{D}}=\varepsilon_{\max}$ achieves low availability, which actually is always around 50% even with more iterations. This is

because the optimal solution of the problem in (14) is obtained when the equality in (16) holds, which cannot be strictly achieved with stochastic optimization. Nonetheless, by reducing ε_{D}, the availability can be remarkably improved with a minor increase of the minimal bandwidth. Besides, the complexity of training for achieving high availability is not high. Less than 10 s is required for 10 000 iterations by a computer with Intel® Core™ i9-9920X CPU without using the acceleration from GPU.

V. Conclusion

In this paper, we proposed an alternative approach of "learning to optimize", where the relation between the solution of constrained-optimization problem and the system parameters are learnt by DNN without using labels for training. The problem can either be variable optimization or functional optimization. We illustrated the approach by considering bandwidth minimization problem under the QoS constraint of URLLC, which is formulated as a variable optimization problem. Simulation results showed that the convergence speed of the proposed unsupervised learning is insensitive to the required reliability, and the solution learnt by DNN can achieve high availability with a minor loss of the bandwidth efficiency.

Appendix A
Proof of the Equivalence between Problems (1) and (2)

Proof: Denote the optimal solution of problem (1) for each value of $\boldsymbol{\theta}$ as $\boldsymbol{x}^*(\boldsymbol{\theta})$, and denote the objective function of problem (2) as $\mathcal{F}\left(\boldsymbol{x}(\boldsymbol{\theta})\right) \triangleq \int_{\boldsymbol{\theta}\in\mathcal{A}} f\left(\boldsymbol{x}(\boldsymbol{\theta});\boldsymbol{\theta}\right) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}$. Since problems (1) and (2) have the same constraints, they have the same feasible area. Therefore, $\boldsymbol{x}^*(\boldsymbol{\theta})$ is feasible for problem (2). Then, in order to prove the equivalence of the two problems we only need to prove that for each feasible solution of problem (2), $\boldsymbol{x}_0(\boldsymbol{\theta})$, we have $\mathcal{F}\left(\boldsymbol{x}^*(\boldsymbol{\theta})\right) - \mathcal{F}\left(\boldsymbol{x}_0(\boldsymbol{\theta})\right) \leq 0$.

Since $\boldsymbol{x}_0(\boldsymbol{\theta})$ is also feasible for problem (1) and $\boldsymbol{x}^*(\boldsymbol{\theta})$ is optimal for problem (1), we have,

$$f\left(\boldsymbol{x}^*(\boldsymbol{\theta});\boldsymbol{\theta}\right) - f\left(\boldsymbol{x}_0(\boldsymbol{\theta});\boldsymbol{\theta}\right) \leq 0, \ \forall \boldsymbol{\theta} \in \mathcal{A}. \tag{A.1}$$

Considering that $p(\boldsymbol{\theta}) > 0$, we further have,

$$\begin{aligned} &\mathcal{F}\left(\boldsymbol{x}^*(\boldsymbol{\theta})\right) - \mathcal{F}\left(\boldsymbol{x}_0(\boldsymbol{\theta})\right) \\ &= \int_{\boldsymbol{\theta}\in\mathcal{A}} \left[f\left(\boldsymbol{x}^*(\boldsymbol{\theta});\boldsymbol{\theta}\right) - f\left(\boldsymbol{x}_0(\boldsymbol{\theta});\boldsymbol{\theta}\right)\right] p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \leq 0. \end{aligned} \tag{A.2}$$

This completes the proof. ■

References

[1] E. Zeidler, *Nonlinear functional analysis and its applications: III: variational methods and optimization*. Springer Science & Business Media, 2013.

[2] S. Boyd and L. Vandanberghe, *Convex Optimization*. Cambridge University Press, 2004.

[3] 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. on Signal Proc.*, vol. 66, no. 20, pp. 5348–5453, Oct. 2018.

[4] L. Liu, B. Yin, S. Zhang, X. Cao, and Y. Cheng, "Deep learning meets wireless network optimization: Identify critical links," *IEEE Trans. on Network Science and Engineering*, pp. 1–1, 2018.

[5] J. Guo and C. Yang, "Predictive resource allocation with deep learning," in *IEEE VTC Fall*, 2018.

[6] 3GPP, *Study on Scenarios and Requirements for Next Generation Access Technologies*. Technical Specification Group Radio Access Network, Technical Report 38.913, Release 14, Oct. 2016.

[7] A. Aijaz, M. Dohler, A. H. Aghvami, *et al.*, "Realizing the tactile internet: Haptic communications over next generation 5G cellular networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 82–89, Apr. 2017.

[8] C. She, C. Yang, and T. Q. S. Quek, "Cross-layer optimization for ultra-reliable and low-latency radio access networks," *IEEE Trans. on Wireless Commun.*, vol. 17, no. 1, pp. 127–141, Jan 2018.

[9] O. C. Zienkiewicz, R. L. Taylor, P. Nithiarasu, and J. Zhu, *The finite element method*. McGraw-hill London, 1977, vol. 3.

[10] J. Gregory, *Constrained optimization in the calculus of variations and optimal control theory*. Chapman and Hall/CRC, 2018.

[11] D. G. Luenberger, *Optimization by vector space methods*. John Wiley & Sons, 1997.

[12] K. Hornik, M. B. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[13] S. Schiessl, J. Gross, and H. Al-Zubaidy, "Delay analysis for wireless fading channels with finite blocklength channel coding," in *ACM MSWiM*, 2015.

[14] W. Yang, G. Durisi, T. Koch, *et al.* , "Quasi-static multiple-antenna fading channels at finite blocklength," *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 4232–4264, Jul. 2014.

[15] M. Condoluci, T. Mahmoodi, E. Steinbach, and M. Dohler, "Soft resource reservation for low-delayed teleoperation over mobile networks," *IEEE Access*, vol. 5, pp. 10 445–10 455, May 2017.

[16] L. Liu, P. Parag, J. Tang, W. Y. Chen, and J. F. Chamberland, "Resource allocation and quality of service evaluation for wireless communication systems using fluid models," *IEEE Trans. on Inf. Theory*, vol. 53, no. 5, pp. 1767–1777, May 2007.

[17] J. Tang and X. Zhang, "Quality-of-service driven power and rate adaptation over wireless links," *IEEE Trans. on Wireless Commun.*, vol. 6, no. 8, pp. 3058–3068, August 2007.

[18] C. She, C. Yang, and L. Liu, "Energy-efficient resource allocation for MIMO-OFDM systems serving random sources with statistical QoS requirement," *IEEE Trans. on Commun.*, vol. 63, no. 11, pp. 4125–4141, Nov 2015.

[19] L. Bottou, "Online algorithms and stochastic approximations," in *Online Learning and Neural Networks*, D. Saad, Ed. Cambridge, UK: Cambridge University Press, 1998, revised, Oct. 2012. [Online]. Available: http://leon.bottou.org/papers/bottou-98x