Machine Learning in the Air

Deniz Gündüz, *Senior Member, IEEE*, Paul de Kerret, Nicholas D. Sidiropoulos, *Fellow, IEEE*, David Gesbert, *Fellow, IEEE*, Chandra R. Murthy, and Mihaela van der Schaar, *Fellow, IEEE*

Abstract— **Thanks to the recent advances in processing speed, data acquisition and storage, machine learning (ML) is penetrating every facet of our lives, and transforming research in many areas in a fundamental manner. Wireless communications is another success story – ubiquitous in our lives, from handheld devices to wearables, smart homes, and automobiles. While recent years have seen a flurry of research activity in exploiting ML tools for various wireless communication problems, the impact of these techniques in practical communication systems and standards is yet to be seen. In this paper, we review some of the major promises and challenges of ML in wireless communication systems, focusing mainly on the physical layer. We present some of the most striking recent accomplishments that ML techniques have achieved with respect to classical approaches, and point to promising research directions where ML is likely to make the biggest impact in the near future. We also highlight the complementary problem of designing physical layer techniques to enable distributed ML at the wireless network edge, which further emphasizes the need to understand and connect ML with fundamental concepts in wireless communications.**

Index Terms— **Autoencoders, channel coding, channel estimation, data-driven methods, distributed learning, distributed resource allocation, deep learning, federated edge learning, joint source-channel coding, machine learning, stochastic approximation, wireless communications.**

I. Introduction

RECENT advances in machine learning (ML) have caused a wave that has swept across all walks of science and engineering. The main premise of ML is to enable computers to learn and perform certain tasks (e.g., classification and prediction) without being explicitly programmed to do so. This is achieved by training algorithms on vast amounts of data available for the task to be accomplished. While the basic ideas and ambitions of ML go back to the 1950s, recent years have witnessed an unprecedented surge in interest in this area, fuelled by the availability of increasingly powerful computers, large and well-curated datasets, and developments in theoretical understanding of various learning algorithms.

Manuscript received April 24, 2019; accepted May 21, 2019. Date of current version September 16, 2019. The work of D. Gündüz was supported by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Program Starting Grant BEACON under Grant 677854. The work of P. de Kerret and D. Gesbert were supported by the ERC under the European Union's Horizon 2020 Research and Innovation Program under Grant 670896. The work of N.D. Sidiropoulos was supported in part by NSF under Grant CIF-1525194, Grant ECCS-1608961, and Grant ECCS-1807660. The work of C. Murthy was supported by the Young Faculty Research Fellowship from the Ministry of Electronics and Information Technology, Government of India. *(Corresponding author: Deniz Gündüz.)*

D. Gündüz is with the Information Processing and Communications Laboratory, Department of Electrical and Electronics Engineering, Imperial College London, London SW7 2BT, U.K (e-mail: d.gunduz@imperial.ac.uk).

P. de Kerret and D. Gesbert are with the Communication Systems Department, EURECOM, 50193 - 06904 Sophia Antipolis, France.

N. D. Sidiropoulos is with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 USA.

C. R. Murthy is with the Department of Electrical and Computer Engineering, Indian Institute of Science, Bengaluru 560012, India.

M. van der Schaar is with the Department of Electrical and Computer Engineering, University of California at Los Angeles, Los Angeles, CA 90095-1594 USA.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2019.2933969

Arguably, the most impressive success stories of modern ML are due to the remarkable efficacy of deep learning, in the form of deep neural networks (DNNs), generative adversarial networks (GANs), and the resurgence of (deep) reinforcement learning ((D)RL) [1], [2]. These tools have resulted in remarkable advances in audio and image recognition, natural language processing, recommender systems, and have beaten human grandmasters in chess and Go. They have also led to many advances in applications from healthcare to autonomous driving, finance, marketing and robotics. The success of these approaches in many practical applications, and particularly the fact that they perform far better than disciplined approaches based on sound theory, has challenged the very foundation of our engineering education. Very few people believed that such a resurgence of 'black box' methods would ever happen, much less that they would work so remarkably well in practice. Hence the latest wave in ML caught many communications engineers by surprise. But because we are engineers, we cannot look away from something that works. We have to understand it and use it to our advantage when possible. ML is certainly 'in the air', with many special issues, workshops, special sessions and panels, exploring the potentials and promises of ML for wireless systems. While the activity in this area is growing at an exponential rate, some seasoned researchers in the community are skeptical, and the impact of ML techniques in practical communication systems and standards is indeed yet to be seen.

Before we go into the challenges of applying ML in wireless systems, we would like to understand whether ML is really a novelty to communications researchers. Is it a completely new paradigm that can transform communications research and future communication systems, or is it yet another "old wine in a new bottle", presenting various old and known techniques with a new flavour? Indeed, the connections between ML and the theory of information transmission and storage are numerous and often striking. The fundamental problem of communication, as stated by Shannon [3], "reproducing at one point either exactly or approximately a message selected at another point," can in fact be recast as a classification problem. More specifically, symbol and sequence detection that constitute the core of any communication system are special cases of the general classification problem, which is at the heart of ML. Shannon's entropy, mutual information, and Kullback-Leibler divergence are widely used in ML as training objectives. Vector quantization (VQ) is key for source coding and rate-distortion, going back to Shannon [4]. VQ is also known as k-means clustering – a staple of unsupervised ML. Universal source coding inherently learns the distribution of the underlying information source in an online fashion [5], [6], and the most successful lossless compression algorithms are based on information theoretic principles, such as Lempel-Ziv and Burrows-Wheeler transforms, and have been successfully implemented for everyday use (gzip, pdf, GIF, etc.). Channel estimation is the task of learning a linear system in a supervised fashion when training/pilots are used. When (e.g., power amplifier) nonlinearities come into play, one has to learn a more general nonlinear system. Coding can be considered as controlled dimensionality expansion from the reduced-dimension *latent* information symbols to the channel input space, and decoding reduces things back to the original low-dimensional information space.

Despite all the fascinating connections, there remain some key differences between generic ML and conventional wireless communication systems. Perhaps the most crucial differences are that i) in communications we have a fairly good grasp of what to expect by way of channel and system models, which obey physical laws; and ii) we have *complete* control of what, when, and how to transmit. In principle this makes communications an overall better playing field for model-based solutions than generic ML applications.

The most striking aspect of the recent success of ML is its 'data-driven' nature – we have access to a lot of data nowadays; hence, we can rely on data to draw conclusions, and design systems like never before. The data-driven approach of ML is significantly different from the model-based approaches that have long dominated communication system design. Communication and networking engineers for many years have developed models with ever-increasing complexity and accuracy for the underlying physical communication channels, antenna patterns, data traffic, user mobility, interference, and many other aspects of communication systems. They have then designed highly complex modulation/ demodulation techniques, error correction codes, and networking protocols based on these models, which can be implemented efficiently (even on low-complexity and energy-limited mobile devices), and can enable reliable communications at fairly high data rates. The model-based approach has been tremendously successful for communication system design, taking us from the first to the fifth generation (5G) of wireless networks, successfully keeping up with the rapidly growing demand for higher quality and lower latency content delivery. However, as we move towards implementing 5G networks and adopting a more flexible network architecture (network function virtualization, software defined networking, etc.), it is likely that there will be many scenarios in which the modeling assumptions used in traditional designs become questionable. For example, with network slicing and multiple service classes, the interference can become highly non-stationary and non-Gaussian. Also, low-latency communications that should be supported with 5G may not allow accurate channel estimation, and short blocklength codes cannot benefit from the ergodicity of some of the randomness in the channel. Similarly, low latency requirements make the structured and modular layered network architecture highly suboptimal, requiring more integrated cross-layer designs, which increases the complexity of optimizing and operating these networks. These challenges point to the need for less structured solutions that are robust to model mismatches. Can the data-driven approach of ML be useful for designing such wireless communication systems and protocols? Modern ML techniques can help us make inferences and predictions about network traffic, user behaviour, application requirements and security threats, all of which can be used for better resource provisioning and improved network operation. 3GPP has already introduced the network data analytics function (NWDAF) in order to standardize the way such data is collected and communicated across various network functions [7]. While this has limited functionality at the moment, it is widely accepted that analysis using higher layer network and user behaviour data will be an integral part of 5G and future communication network architectures, where network functions will interact through NWDAF to provide relevant data to be used by other network functions, and will apply various ML techniques on the available data to make control and resource allocation decisions. Therefore, an important question in this context is what type of lower layer data can be used to improve the utilization of limited physical layer resources, and which ML techniques would provide timely and useful inferences and predictions based on this data.

In this paper we will try to answer these questions, focusing mainly on some exemplary applications of ML tools for lower layer design. We note here that the goal of this paper is not to provide a survey of recent results in this very active research area, but to highlight some of the striking recent results that promise significant gains compared to conventional physical layer design techniques, and provide a general discussion on why these techniques are promising, and the potential roadblocks for their implementation in real systems and adoption in standards. We refer the readers to excellent survey and overview papers on various aspects of ML in wireless communications to gather a more complete picture of its recent applications in different settings [8]–[13]. Next, we will go over some of the major challenges of applying ML in the lower layers of the protocol stack.

A. Challenges of Applying ML Tools in Wireless Communications

A major criticism for the data-driven approach to communication system design is the 'black-box' nature of some of the ML algorithms, e.g., DNNs, and the lack of guarantees for performance; whereas communication engineers are accustomed to providing performance guarantees on error probability, interference level, channel outage, latency, etc. In many cases, such as emergency communication networks or for critical infrastructures, reliability and latency requirements can be extremely stringent. However, such provable guarantees hinge on the assumed channel, traffic, and device models, and their validity is as good as the accuracy of these models. Channel modeling, for example, no matter how ingenious, is always approximate, and the true channel dynamically evolves and is subject to all sorts of nonlinear/phase transition effects, from amplifier nonlinearities to loss of synchronization, which bring us closer to the realm of more general ML. On the contrary, the data-driven approach does not need powerful models, and instead can learn the optimal system architecture simply from available data. One particularly striking example is the use of the autoencoder as a general nonlinear detection mechanism – without having to physically model, estimate, and explicitly implement an equalizer or an error control mechanism. The advantage of such an approach is that it can "invert" even unknown nonlinear channels directly, based only on training data and nothing else. Therefore, it is not clear which would provide a more reliable communication system: the one optimally designed based on complex yet approximate models, or the one designed by black-box ML algorithms based on training data. While the former is limited by the accuracy of the model in representing the reality, the latter uses real data in the learning process, but the data is always limited in size and generalizability. We expect that ML-based solutions will be effective particularly when an accurate model for the problem of interest is not available (i.e.,'model-deficit' as referred to in [13]), and a sufficiently large and representative training dataset is available.

The 'black box' aspect of DNNs also brings along the *interpretability* problem. Understanding the reasons behind the success or failure of ML methods, particularly those based on DNNs, is an on-going research challenge [14], which is yet to be addressed satisfactorily. From an engineering perspective, not knowing the reasons behind the decisions taken by an algorithm makes it very difficult to tackle failures, or to predict the impact of changes in the environment on the performance. Also relevant for communication networks is to guarantee some sense of fairness across users, such that they are not penalized unintentionally by a ML algorithm due to the type of their device, their location, protocol being used, etc. Fairness in ML is a very important and growing research challenge, particularly for applications that involve personal data, and those that make decisions that have direct impact on our lives, e.g., automatic evaluation and ranking of CVs, recommender systems for online shopping, and even for criminal investigations [15]. Given the sensitivity of our mobile traces, and the close association between users and their mobile devices, fairness is likely to be an important concern in future application of ML in wireless systems as well.

Another challenge of applying data-driven ML tools to wireless systems is the limited availability of training data. Unlike in computer vision, speech processing, or health-care applications, in most wireless applications standardized datasets for testing and comparison of proposed ML techniques are not available. However, we expect that, with the increasing adoption of ML techniques, more public datasets will become available to the community. There are a number of initial efforts in this direction, and several publicly available datasets can be used to perform and compare some basic ML tasks on wireless signals [16]–[19].

Even if such datasets become available, it is questionable whether success on such datasets can promise success in other channel and network conditions. Wireless channels are often highly non-stationary, and offline training on a generic dataset may not lead to satisfactory results when tested on a very different wireless environment. This may require online training of the existing models to adapt them to the current scenario; however, training time of an ML model for reasonable performance is often beyond the operation timescales of communication systems.

On the other hand, when a reasonably accurate model (e.g., for the underlying communication channel) is available, one can generate synthetic data from the model, which can be used for offline training. In this case, by a judicious choice of the architecture, one can arrive at an ML algorithm that (at least empirically) outperforms its conventional counterparts, when algorithms of similar computational complexity are compared. This points to another setting in which ML-based techniques have proven useful: even when the system model is accurately or perfectly known, the optimal solution may be too complex, or even intractable [20] (referred to as 'algorithm-deficit' in [13]). In such a case, the model can be used to generate data, which can then be used to train a limited-complexity ML-model, which can either try to imitate an available model-based approximate or optimal solution, or directly achieve the optimal performance. Such an approach has been shown to provide approximate solutions to even NP-hard problems using moderate computational resources [21], [22], or to outperform human experts in fully known yet highly complex models, such as chess, Go, or Atari games [23], [24]. We will provide more details in Section V along with several examples how ML can be used in wireless networks with known models as a way to optimize the network performance (e.g., sum rate in a multi-user network).

Another issue raised when adopting ML-based techniques in wireless communication networks is the limited computational and memory resources available to most wireless devices, especially low-complexity terminals at the network's edge. Many of the impressive results with data-driven ML techniques are obtained using very powerful computing machinery and massive datasets, which may not be possible to reach by mobile devices with limited computation, memory, and energy resources. The computing power of even the most recent mobile devices is orders of magnitude less than high performance computers used to train complex ML models. Also, each wireless end-user device typically has only a limited amount of data, further limiting the training capabilities. The current approach to overcome the limitations of wireless devices is cloud or edge processing, in which all the data available at wireless devices are transferred to an edge or cloud unit, where a powerful ML algorithm can be centrally trained using all the data. However, such a solution comes at the cost of transferring the data from energy and bandwidth limited wireless edge devices to a central edge or cloud processor, and the latency this would incur – not to mention privacy concerns, which are increasingly becoming a serious challenge to centralized data processing. This necessitates new ways of achieving decentralized learning in the wireless setting.

Decentralized learning and decision making is ultimately limited by how much information is let to flow between the learning devices, and how much noise corrupts the local device's information. Clearly, algorithms which can adapt to arbitrarily distributed information settings would be highly desirable. More details and concrete examples will be given in Section VI on distributed ML at the wireless network edge.

In the rest of this paper, we will highlight some specific problems in wireless communication networks, and in particular at the physical layer, where we believe ML techniques can make a significant impact. In some of the settings, data-driven solutions are posed to solve hard wireless networking problems, trained on data generated from existing models. These emphasize the use of ML as an optimization technique to obtain solutions that can surpass the state-of-the-art. We will also highlight some applications in which data-driven ML techniques are suitable due to the lack of accurate models. Although we provide pointers to a large number of key references, the presented examples are naturally influenced by our personal research experiences and interests. Nonetheless, we believe that the highlighted observations are likely to 'generalize' to other relevant problems and scenarios.

II. Deep Learning Based Detection and Decoding

Data detection over a noisy channel, which is an essential component of any communication system, is inherently a classification problem. Current detection systems rely on model-based solutions, employing a mathematical model describing the underlying communication channel. Moreover, we typically use a detector derived assuming perfect channel state information (CSI) at the receiver, with the channel state replaced by its estimate computed from training symbols. This renders the detector sub-optimal in the presence of CSI estimation errors, and a well-trained ML algorithm can outperform classical approaches. In the context of data detection under a Poisson channel model (which arises in molecular communication), in [25], a recurrent NN (RNN) is used in the presence of intersymbol interference (ISI). While the proposed RNN structure can be trained to learn to disentangle the impact of ISI without any additional information, the performance of the classical Viterbi decoder (VD) depends heavily on the accuracy of CSI, as well as the memory length of ISI in the channel. In [25], authors also train a detector based purely on data collected from a molecular communication channel. Since accurate models for this system are lacking, they show that, under CSI estimation errors, the NN-based detector performs significantly better than state-of-the-art detectors. This result corresponds to a fully data-driven approach for a complex system with hard-to-model imperfections and nonlinearities, where ML provides an attractive alternative.

The detection problem is also studied in [26] considering a MIMO channel:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w}, \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^N$ is the received vector, $\mathbf{H} \in \mathbb{R}^{N\times K}$ is the channel matrix, $\mathbf{x} \in \{-1, +1\}^K$ is the unknown channel input vector consisting of independent and equally likely binary symbols, and $\mathbf{w} \in \mathbb{R}^N$ is the noise vector consisting of independent zero-mean Gaussian variables of unknown variance. Even under the perfect knowledge of the channel model and the channel matrix $\mathbf{H}$, as the dimensionality of the problem, $N \times K$, increases, the optimal maximum likelihood (ML) detector becomes impractical due to formidable computational complexity. The authors propose a DNN-based detector. The challenge here is to find the best way to feed the CSI to the DNN to allow the network to learn to exploit this additional information. The authors exploit the structure of a projected gradient solution, and design the network architecture accordingly. In particular, they feed into each layer of the network $\mathbf{H}^T\mathbf{y}$, $\mathbf{v}_k$, $\mathbf{x}_k$ and $\mathbf{H}^T\mathbf{H}\mathbf{x}_k$, where $\mathbf{v}_k$ and $\mathbf{x}_k$ are obtained iteratively as outputs of each layer of the DNN. The results in [26] show that the DNN-based decoder achieves comparable performance with a high-complexity decoder based on semi-definite relaxation, while running 30 times faster. This is a good example of a solution that combines DNN-based "black-box" solution with domain knowledge, which is used to "steer" the DNN to exploit the available information in an efficient manner. Note that, the decoder is not provided any additional information, and in theory should be able to learn to mimic this structure; however, providing the same information in the most convenient form can speed up the learning process, and avoid suboptimal local optima.

In [27], the authors study the data detection problem over a known channel model, without CSI at the receiver. The DNN-based decoder in this case is trained to output the estimated data symbols based purely on the received signal, without explicitly estimating the wireless channel. The DNN is trained using synthetically generated input-output data. Specifically, the channel is drawn from the wireless world initiative for new radio (WINNER II) model, which models a typical urban channel with 24 sample-spaced paths. The DNN consists of an input layer, three hidden layers and an output layer, with a heuristically selected number of neurons in each layer. The rectified linear unit (ReLU) is used as the activation function in all but the output layer, where the sigmoid function is used to map the output to the interval $[0, 1]$. MSE between the transmitted and predicted symbols is used as the loss function for training the DNN. Numerical results illustrate several interesting points. First, when sufficiently many pilots are present, the bit error rate (BER) performance of the DNN-based decoder matches that of the minimum mean square error (MMSE) receiver. However, under non-ideal conditions, such as fewer pilots, absence of the cyclic prefix, or nonlinear clipping noise, the DNN-based decoder can significantly outperform the MMSE receiver. Of course, the MMSE receiver is no longer optimal under these non-idealities, and another hand-crafted solution that address them could perform as well or better than the DNN-based decoder. Nonetheless, the deep learning approach offers a relatively straightforward and promising solution, that can potentially deal with a variety of non-idealities in a robust manner. In [28], the authors carry this idea further and illustrate over-the-air results of an online trainable OFDM receiver.

A. Learning to Decode

While the above works mainly focus on detecting the channel input symbols, DNNs can also be used to recover coded symbols. Decoding of codewords from a certain channel code is another classification problem. However, the number of classes to classify the received signal into grows exponentially with the blocklength, leading to exponentially growing training complexity. Therefore, most of the current approaches to DNN-based channel decoding incorporate DNNs into the existing decoder structures. For example, [29] uses a NN to learn the weights that should be assigned to the Tanner graph of the belief propagation (BP) algorithm. In [30], authors propose improving the performance of conventional iterative decoding for polar codes by complementing it with NN-based components. In particular, they divide the decoder into subblocks, each of which is replaced by a NN-based decoder, and the results of these decoders are fed into a BP decoder. This allows controlling the training complexity by adjusting the number of subblocks.

A fully DNN-based channel decoder is considered in [31]. To keep the complexity reasonable, codelength is limited to 16 while the code rate is fixed to $1/2$. The authors trained the decoder NN both for a polar code and a random code. While a performance close to a maximum aposteriori (MAP) decoder can be achieved for the polar code, the gap to the MAP decoder performance is much larger for the random code. Although this gap can be closed with increasing the number of training epochs, the result highlights the point that NNs are most effective when the data has an underlying structure that can be learned. The authors also considered limiting the set of codewords observed during training. This is to test whether the NN-based decoder can generalize to unseen codewords. They observed that this was indeed the case for the polar code; the decoder was able to learn to decode codewords it has never seen before, which can only be explained by the fact that the NN-based decoder has learned the structure of the decoding algorithm. This is not the case for the random code, which did not have any particular structure that could be exploited by the NN-based decoder.

B. Observations

Certain features that are common to the aforementioned works are worth mentioning. First, most of them use the so-called one-hot representation of the transmitted signal [32], [33]. In the one-hot representation, the signal is represented as a binary vector of length equal to the number of possible signals. The binary vector contains a single 1 at the location corresponding to the transmitted signal, and zeros everywhere. While one-hot encoding typically provides better results as it prevents any numerical ordering between the inputs, it also leads to an exponentially growing input size for channel decoding.

The output layer of the DNN that attempts to reconstruct the input signal is typically chosen as the sigmoid function. In this case, the DNN attempts to output the likelihoods of possible signals, which is useful, for example, in detection of coded symbols, where the bit log likelihood ratios need to be fed into a channel decoder.

Another interesting recent development is the use of RNNs with long-short term memory (LSTM), which allows for smaller generalization error [25]. This allows for unseen channel instantiations to be handled effectively.

III. Deep Learning for Channel Estimation and Channel State Information (CSI) Feedback

CSI is essential in wireless communications. On the receiver side, knowledge of the channel state allows coherent detection and decoding. Channel estimation at the receiver is typically carried out by sending pilots from the transmitter. On the transmitter side, CSI allows employing adaptive transmission techniques, which can provide significant gains in performance and efficiency. Transmitter CSI is especially important in massive MIMO systems. For time division duplex schemes CSI can be obtained at the transmitter side by exploiting reciprocity. However, in frequency division duplex schemes, CSI estimated at the receiver needs to be conveyed to the transmitter over a feedback link. In order to minimize the resources dedicated to CSI feedback, it is essential to compress the CSI estimate at the receiver as efficiently as possible. Below we review some of the recent applications of DNNs to the channel estimation and CSI compression problems. While detection and decoding studied in the previous section correspond to classification problems, channel estimation is a regression problem, and CSI compression represents an unsupervised clustering problem.

A. Channel Estimation

Recall that MMSE channel estimation entails knowledge of the channel statistics and a potentially computationally expensive conditional mean computation. In [34], the authors model the channel as conditionally Gaussian distributed given a set of (hyper)parameters. These hyperparameters are also random, whose distribution is eventually learned from training data. The MMSE estimator under this model can be written as a linear estimator, with weights depending on the statistics of the hyperparameters. By vectorizing the MMSE estimate, the authors write the estimator in a form that is amenable to implementation as a feed-forward neural network with two linear layers connected by a nonlinear activation function. These layers are made learnable, and are trained via stochastic gradient descent with the mean squared channel estimation error as the loss function. It is shown that, under certain assumptions, this can lead to a computationally inexpensive, near-optimal MMSE estimator when the channel covariance matrix is Toeplitz and has a shift-invariance structure. Simulation results suggest that the NN based channel estimator outperforms state-of-the-art estimators and has low complexity.

In the context of wideband channels, [35] models the channel time-frequency response as an image, and the pilot based samples as a low-resolution sampled version of the image. The authors use convolutional neural networks (CNN) based image super-resolution and image restoration techniques to estimate the channel, with the mean squared error (MSE) as the loss function. Empirically, the performance is demonstrated to be similar to that of an ideal MMSE estimator that has perfect knowledge of the channel statistics.

As a final note on DNN-based channel estimation, above ideas have been extended to the case where the receiver has fewer RF chains than antenna elements, for example, in mmWave systems. In this case, the key challenge for the receiver is to estimate the channel from compressed measurements. In [36] and [37], it is shown that these estimators can even outperform estimators based on sparse signal recovery, when trained with sufficient amount of data.

B. CSI Compression

As mentioned above, accurate CSI knowledge at the transmitter can significantly increase the performance of wireless communication systems, for example, by avoiding poor channel states, or by employing beamforming. In frequency duplex systems, the transmitter depends on feedback from the receiver to acquire CSI. In order to limit the resources dedicated to CSI feedback, it is important to design an efficient compression algorithm which can provide a high accuracy CSI estimate to the transmitter while using limited communication resources, measured in terms of bits per channel symbol. This becomes particularly important in massive MIMO systems, which require accurate downlink CSI to achieve the promised performance gains, while the CSI feedback overhead can be excessive due to the massive number of antennas.

Simple scalar quantization methods are not suitable for schemes that are highly sensitive to CSI estimation quality at the transmitter. Moreover, they cannot exploit the spatial structure in the channel matrix, and result in high feedback overhead. CSI feedback reduction techniques based on vector quantization [38] are also limited, particularly for massive MIMO systems, as the codebook size, and thus, grow proportionally with the number of transmit antennas. More recently, compressive sensing has been considered to exploit the sparse structure of the underlying channel in a transform domain [39], [40]. While this provides significant reductions in CSI feedback, they do not fully exploit the correlations among antennas.

Since CSI compression is a special case of the more general data compression problem, let us briefly mention here how ML techniques can be used for data compression in general. Data compression is a fundamental problem in information and coding theory, and significant research efforts have been dedicated to developing efficient compression algorithms for various information sources, such as image, audio, or video. The traditional approach has been to leverage expert feature knowledge for each domain to design specific compression schemes; so much so that there have been separate research communities working on each of these data domains, and distinct compression standards, such as MP3, JPEG and MPEG, have been developed. In most cases the algorithms try to exploit the sparsity of the information source in a transform domain, such as discrete cosine transform in image compression, or some other structures, such as motion compensation in video compression. While these highly specialized techniques, which have been refined and perfected over many decades of research and development, provide reasonably good performance in general, recently there has been significant progress in exploiting DNN architectures for compression in all these data domains [41]–[44], with results meeting or surpassing state-of-the-art expert-based compression techniques, which is quite remarkable.

The main component in most of these implementations is the autoencoder structure. An *autoencoder* is a pair of NNs, called the *encoder* and the *decoder* networks. The output of the encoder network, called the *bottleneck layer*, is the input to the decoder network. The two networks are trained jointly with the goal of recovering the input at the output of the decoder. Typically the bottleneck layer has lower dimension than the input data, and if the autoencoder can learn to recover the input with a minimal distortion, this means that the bottleneck layer carries the essential information to approximately reconstruct the input data; and hence, can be considered as a compressed version of the input signal. Autoencoders are used in ML for feature extraction or as generative models for data [1]. It is an unsupervised learning technique as it does not require any labels.

The main advantage of autoencoders for data compression is that they do not require the knowledge of the underlying data distribution, or explicit identification of a certain structure, but instead they learn a low-dimensional representation directly from data. Moreover, autoencoders can be optimized for very specific information sources. While standard image compression techniques apply the same algorithm on all types of images, an autoencoder can be trained only on, say, underwater images, and learn specific features of these images, resulting in a much higher compression efficiency.

This data-driven autoencoder-based compression approach is particularly attractive for CSI feedback compression as it is difficult to identify and characterize the features of channel matrices, which can have quite complicated interdependencies through the physical environment. On the other hand, acquiring CSI data for training can be easy if we have a relatively simple model that can represent the physical channel accurately. Many such models have been developed over the years, such as the 3GPP spatial channel model (SCM) [45], WINNER [46], IEEE 802.16a,e [47], or the more advanced geometry-based COST 2100 stochastic channel model [48].

An autoencoder based compression scheme, called CSINet, is studied in [49], and it shown to provide significant improvement in compression efficiency compared to the state-of-the-art techniques exploiting sparsity. In [50], the authors consider temporal correlations in time-varying channels, and improve the performance of CSINet for this scenario using a RNN. Utilizing channel reciprocity, the authors in [51] use the uplink CSI as additional correlated side information to further improve the compression efficiency.

However, the aforementioned works focus mainly on the dimensionality reduction aspect, and they do not directly tackle the compression problem, which requires a binary representation of the CSI, which is then transmitted reliable over the feedback link. While dimensionality reduction can potentially reduce the required feedback resources, in principle each autoencoder output is still a real number that needs to be quantized before being fed back to the transmitter. In practice, sufficient accuracy can be achieved by using 32-bit quantization for each of the autoencoder outputs. However, it is not clear if this leads to the most efficient binary representation of the CSI matrix. An alternative approach is to directly incorporate the quantization operation into the autoencoder training process. This is challenging, however, as the quantization operation is non-differentiable. Various methods have been proposed in the image compression literature to overcome this difficulty. In [52] quantizer gradient is approximated as 1 in the backward pass, while [42] replaces the quantization with additive uniform noise, and a stochastic binarization function is used in [53].

Quantization is incorporated into the CSI compression architecture in [54], where a more advanced autoencoder architecture is employed compared to [49], and trained together with the quantizer. The output of the quantizer is entropy coded as in standard image compression algorithms to further reduce the compression rate. This leads to significant improvement in the compression efficiency. For example, for 32 transmit antennas and 256 subcarriers, the results in [54] show that the proposed architecture, which also includes the quantization and entropy coding can provide approximately 7dB reduction in the mean-square error of the reconstructed channel matrix at $0.01 - 0.12$ bits per channel symbol.

IV. Autoencoders for End-to-End Communication System Design

The correspondence of the autoencoder structure to a communication system with an encoder and a decoder is quite obvious. As mentioned in Section III-B, autoencoders have been successfully applied to image and video compression, which can be considered as communication over a finite-rate error-free channel. End-to-end learning of encoder and decoder functions for communications over a physical layer channel is first proposed in [55], and later expanded in [56]. The noisy communication channel that connects the output of the encoder NN to the input of the decoder NN is treated as an untrainable layer with a fixed transformation. This end-to-end training of the physical layer bypasses the modular structure of conventional communication systems that consists of separate blocks for data compression, channel coding, modulation, channel estimation and equalization, each of which can be individually optimized. While this modular structure has advantages in terms of complexity and ease of practical implementation, it is known to be suboptimal. An autoencoder is trained for coding and modulation over an additive white Gaussian noise channel in [56], and it is shown to have a performance very close to conventional coding and modulation scheme in short blocklengths.

The aforementioned works on autoencoder-based end-to-end physical layer design assume a known channel model, and the encoder and decoder networks are trained jointly by simulating many realizations of this channel model. While models for wireless channels are considered to be accurate in general, they may still have mismatch with the real channel experienced by the transceivers, limiting the overall performance of the system. An alternative would be to use a GAN architecture to learn a channel model based on real data collected from the channel. This can provide a more accurate model of the channel, particularly if sufficient data can be collected from the channel. In [57], the authors propose to use the learned GAN as the channel layer between the encoder and decoder NNs of an end-to-end communication system.

A fundamental challenge in training autoencoders directly on a real channel is the significant delay this may cause. Since the encoder and decoder must be trained jointly, the back-propagation has to propagate the gradient from the receiver to the transmitter, requiring a feedback link during training, which would significantly slow down training. To circumvent this limitation, [58] proposes a two-phase training approach: the first phase uses a channel model as before and the encoder and decoder are trained based on this model as before. Once these networks are deployed at the transmitter and the receiver, the receiver network is trained further based on the transmission of known signals from the transmitter. This is similar to pilot transmission in channel estimation, and does not require feedback to the transmitter.

A. Joint Source-Channel Coding (JSCC)

All the above works have exclusively focused on transmitting bits over the noisy channel, that is, the goal is to design error correction codes jointly with modulation, channel estimation, etc. Note that, when the input to the encoder is a bit sequence, there is no structure in the data, and the goal is to learn the best mapping of the message bits into the channel input space, and jointly the best inverse mapping. This is achieved mainly by distributing the input signals as much as possible in the channel input space within the constraints of the transmitter, and taking into account the random channel transformation. However, in many real applications, the goal is to transmit some information signal, e.g., a picture, video, or an audio signal, which is not in the form of a sequence of equally likely bits, and typically has significant redundancy.

The current standard approach to transmission of such signals is to first compress them with a source coding algorithm in order to get rid of the inherent redundancy, and to reduce the amount of transferred information; then the compressed bitstream is encoded and modulated over the channel. Shannon's *separation theorem* proves that this two-step separate source and channel coding approach is optimal theoretically in the asymptotic limit of infinitely long source and channel blocks [60]. However, in practical applications, JSCC is known to outperform the separate approach, particularly in short-blocklength and low-SNR regimes. Many emerging applications from the Internet-of-things (IoT) to autonomous driving and to tactile Internet require transmission of high data rate information (image/video, various sensor measurements) under extreme latency, bandwidth and/or energy constraints, which preclude computationally demanding long-blocklength source and channel coding techniques. However, characterizing the optimal JSCC in non-asymptotic regimes has remained an open problem, even for fully known source and channel

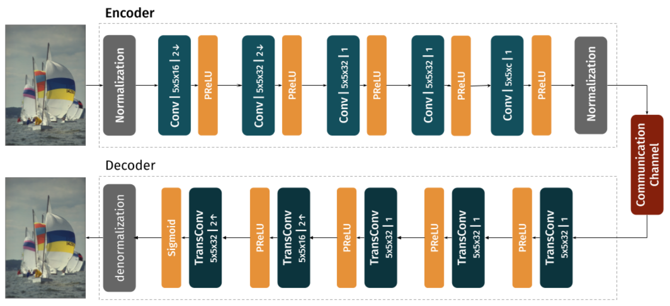

Fig. 1. Encoder and decoder NN architectures used in the implementation of the deep JSCC scheme in [59].

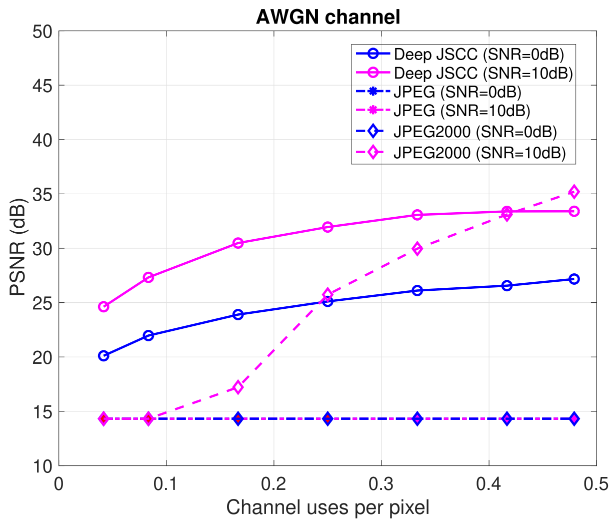

Fig. 2. Performance of the deep JSCC algorithm in [59] on CIFAR-10 test images transmitted over an AWGN channel with respect to the available channel bandwidth per image pixel for different SNR values. For each case, the same SNR value is used in training and evaluation, and a different network is used to obtain each point in the curve.

distributions, and it is significantly more challenging for the transmission of complicated sources, such as images or videos, for which we do not have good statistical models.

Alternatively, a deep JSCC architecture can be trained to map the underlying signal samples directly to channel inputs. Such an architecture is studied for transmission of images over wireless channels in [59]. This can be considered as an "analog" JSCC scheme since, unlike digital systems built upon the separation approach, the input signal is never converted into bits, and the channel input signal is not limited to a finite number of constellation points. The deep JSCC architecture proposed in [59] is illustrated in Fig. 1. This fully convolutional architecture allows compression of images of any size. The results illustrated in Fig. 2 show that deep JSCC outperforms state-of-the-art digital image transmission schemes, e.g., JPEG/ JPEG2000 image compression followed by capacity-achieving channel codes, particularly in the low SNR and small channel bandwidth regimes. Note that, both JPEG and JPEG2000 fail completely at $\mathrm{SNR} = 0$ dB. For $\mathrm{SNR} = 10$ dB, JPEG2000 can provide reasonable quality if the channel bandwidth is sufficiently large. A few aspects of deep JSCC are particularly worth mentioning: First of

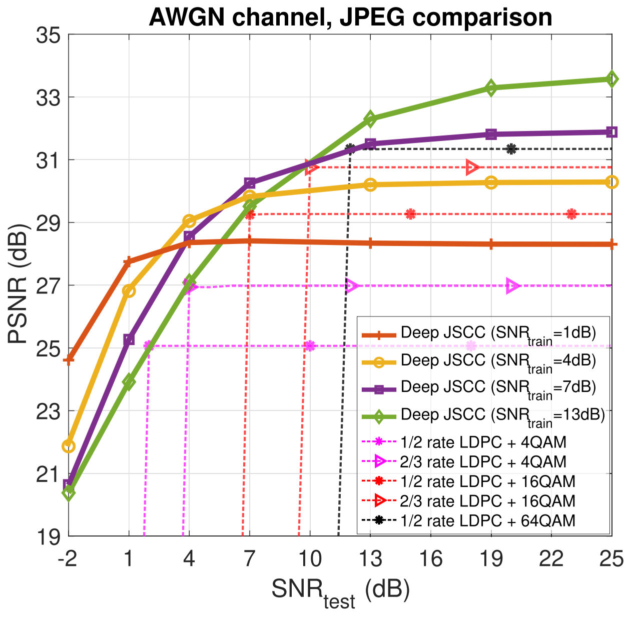

Fig. 3. Performance comparison between the deep JSCC algorithm in [59] and JPEG compression followed by LDPC coding. Deep JSCC is trained on the Imagenet dataset for the indicated $\mathrm{SNR}_{\mathtt{train}}$ values. Compression ratio is $1/12$, i.e., 1 channel use per 12 pixels. $\mathrm{SNR}_{\mathtt{train}}$ value.

all, it provides non-trivial image reconstruction even at very low SNR values and limited channel bandwidths, i.e., in the case of short blocklengths. Moreover, thanks to the analog nature of the encoder, the performance behaves like analog modulation schemes, and exhibits graceful degradation with channel SNR. This can be observed in the performance curves in Fig. 3. A deep JSCC architecture trained for a particular target channel SNR value gracefully degrades if the channel SNR falls below this value, and its performance improves gradually if the channel SNR goes above the target value.

This analog behaviour is particularly attractive for broadcasting to multiple receivers, or when transmitting over a time-varying channel. Indeed, it is shown in [59] that the performance improvement of deep JSCC compared to conventional digital schemes is much higher over fading channels. Note also that, while learning channel codes is challenging even for very limited blocklengths, deep JSCC can achieve performance levels above or comparable with state-of-the-art digital techniques even over large blocklengths.

It is shown in [61] that this deep JSCC architecture also allows bandwidth adaptation through successive refinement; that is, an image can be transmitted over n layers, and a user receiving the first k layers can recover the image with peak signal-to-noise ratio PSNR_k, $k = 1, \ldots, n$. While $\mathrm{PSNR}_1 < \cdots < \mathrm{PSNR}_n$ as expected, PSNR_k is very close to the performance one would obtain if the image was transmitted targeting the total bandwidth available for the first k layers; that is, transmitting the image in layers comes at almost no additional cost, providing seamless bandwidth adaptivity.

V. Machine Learning Based Resource Allocation

An important class of problems where modern ML techniques can help is formed by (NP-) hard resource allocation and decision / scheduling problems – which are very common in wireless communications and networking. Examples range from classical multi-user detection to sum-rate optimal power control, multi-user scheduling, and transmission control – or "smart" data-driven TCP-IP. In the following paragraphs, we will review some illustrating examples.

The case of joint multicast beamforming and antenna selection is considered in [62], where it is shown how a DNN can be used to successfully solve the discrete optimization part of the problem. This is an example of a hybrid strategy, where a DNN is employed to solve part of the problem, synergistically with classical optimization.

Beamforming for minimum outage [63] has also been proven to be NP-hard even when the channel distribution is known *exactly*, and in fact no practically good approximation algorithm was known until very recently. Yet, relying on a sample average 'counting' approximation of outage, simple smoothing, and stochastic gradient updates, a lightweight and very effective algorithm was recently designed in [64] that performs remarkably well, using only recent channel data. The problem is formulated as follows:

$$\min_{\mathbf{w}\in\mathcal{W}}\left\{F(\mathbf{w}) := \Pr\left(|\mathbf{w}^H\mathbf{h}|^2 < \gamma\right)\right\}, \tag{2}$$

where $\gamma > 0$ denotes the outage threshold and $\mathcal{W} \subset \mathbb{C}^N$ is a simple (element-wise or sum) power constraint. We can equivalently express (2) as

$$\min_{\mathbf{w}\in\mathcal{W}} \Pr\left(|\mathbf{w}^H\mathbf{h}|^2 < \gamma\right) \Leftrightarrow \min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_{\mathbf{h}}[\mathbb{1}_{\{|\mathbf{w}^H\mathbf{h}|^2<\gamma\}}]. \tag{3}$$

Define

$$f(\mathbf{w};\mathbf{h}) := \mathbb{1}_{\{|\mathbf{w}^H\mathbf{h}|^2<\gamma\}} = \begin{cases} 1, & \text{if } |\mathbf{w}^H\mathbf{h}|^2 < \gamma \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

as the indicator function of the event $|\mathbf{w}^H\mathbf{h}|^2 < \gamma$. Consider a given set of 'recent' channel realizations $\mathcal{H}_T := \{\mathbf{h}_t\}_{t=1}^T$. Utilizing $\mathcal{H}_T$, we may construct the following sample average estimate of $\mathbb{E}_{\mathbf{h}}[f(\mathbf{w};\mathbf{h})]$

$$\hat{F}(\mathbf{w};\mathcal{H}_T) := \frac{1}{T}\sum_{t=1}^{T} f(\mathbf{w};\mathbf{h}_t). \tag{5}$$

The interpretation is that we minimize the total number of outages over ('recent') channel history - very reasonable, since under appropriate mixing conditions we have

$$\lim_{T\to\infty} \hat{F}(\mathbf{w};\mathcal{H}_T) = \mathbb{E}_{\mathbf{h}}[f(\mathbf{w};\mathbf{h})] = F(\mathbf{w}), \quad \forall\, \mathbf{w}\in\mathcal{W}, \tag{6}$$

almost surely. Replacing $F(\mathbf{w})$ by $\hat{F}(\mathbf{w};\mathcal{H}_T)$ in (2), we obtain

$$\min_{\mathbf{w}\in\mathcal{W}} \hat{F}(\mathbf{w};\mathcal{H}_T). \tag{7}$$

The final step is to construct a smooth approximation of $f(\mathbf{w};\mathbf{h})$, and optimize the resulting function using stochastic gradient descent. As it is shown in [64], this approach works unexpectedly well, on a problem that has challenged many disciplined optimization experts for years.

Finally, the sum-rate optimal power control problem is known to be NP-hard, but we have good, albeit computationally expensive, approximation schemes at our disposal. These include the iterative weighted minimum mean squared error (WMMSE) approach [65], [66], and successive convex approximation [67]. These algorithms are too complex for practical implementation, but a key idea advocated in [20], [68] is that we can take this complexity offline by training a DNN to mimic the input-output behavior of the WMMSE algorithm. The way to do this is to use historical (measured) and/or simulated channel data, run the WMMSE algorithm offline to generate the associated power allocation values, and use these input-output pairs to train the DNN. At run time, we simply pass the input through the trained neural network, which is far cheaper than running WMMSE online, and works remarkably well.

The approach can be further refined by training the network to optimize sum rate directly as described in [69]. In that case, the sum rate is directly differentiated with respect to the coefficients of the DNNs, which allows to further improve the performance. The existing state-of-the-art solutions and the approximation approach described above are still used to initialize the optimization, and hence avoid the inefficient local optima.

A. ML for Decentralized Resource Allocation

While the solution approach to the resource allocation problems above exploits a central common intelligence with full knowledge of the network state, many networking problems require decentralized optimization. Such settings include, for instance, coordination and cooperation tasks among radio devices in the absence of a central controller, and can be linked to so-called *team decision (TD)* and *decentralized control* problems, which are notoriously difficult to tackle. In TD problems, multiple-agents aim at cooperating to achieve a common goal on the basis of imperfect and nonhomogeneous information.

The derivation of robust multi-device decision-making algorithms with arbitrary input uncertainties across agents is well known to be a challenging task, and cannot be solved via conventional optimization methods. Team decision problems were first formulated by Radner in [70], and later studied by Marschak and Radner in [71]. Although some particular simple cases could be solved (e.g., a linear objective), the general problem remains open, with no good approximate solution. This makes this class of communication design problems an interesting playing field for ML.

Distributed radio resource optimization in communication networks with the goal of maximizing the network performance can be recast as a multi-agent coordination problem. Typically, the agents (i.e., radio devices) optimize their transmission parameters on the basis of imperfect local information, for example, noisy CSI [72].

Consider the general problem with n agents, where agent j takes decision $\boldsymbol{d}_j$ using the information locally available, denoted by $\mathbf{y}_j$:

$$\boldsymbol{d}_j = \boldsymbol{s}_j(\mathbf{y}_j), \tag{8}$$

where $\boldsymbol{s}_j$ can be any arbitrary function from the information space to the decision space. Information $\mathbf{y}_j$ available at agent j

may be the result of sensing, estimation, feedback, or information shared over the backhaul network before the actual transmission. Due to the limited amount of available resources, the resulting estimate obtained is expected to be a potentially imperfect and/or incomplete estimate of the true representation of the network state $\mathbf{x}$. This information model is extremely general and encompasses as special cases the *centralized* CSI configuration of the problems discussed above as well as the local CSI configuration.

We focus here on the fully cooperative scenario, in which all the agents aim at jointly maximizing a common objective function u in an expected sense. The optimization problem can then be written as

$$(\boldsymbol{s}_1^\star,\ldots,\boldsymbol{s}_n^\star)=\underset{\boldsymbol{s}_1,\ldots,\boldsymbol{s}_n}{\operatorname{argmax}}\ \mathrm{E}\left[u(\mathbf{x},\boldsymbol{s}_1(\mathbf{y}_1),\ldots,\boldsymbol{s}_n(\mathbf{y}_n))\right], \quad (9)$$

where the expectation is taken over the joint distribution $p_{\mathbf{x},\mathbf{y}_1,\ldots,\mathbf{y}_K}$. We assume that all the agents know this distribution, or equivalently, as it will become clear later on, that the training dataset is available at all the agents. It is important to note that we consider for the sake of clarity in (9) an optimization problem with only *implicit* constraints on the decision functions and no *explicit* constraints. Yet, this formulation trivially extends to constrained optimization problems.

This is however a rather simplified case of more general decentralized multi-agent optimization problems as we consider only a one-shot optimization rather than a repeated setting where agents take decisions in multiple rounds, while receiving some form of feedback at the end of each round. The feedback could be in the form of a reward function, or explicit information exchange among the agents, which can be classified as active and passive feedback, respectively. In the case of active feedback, each agent would optimize the information to share with the other agents, possibly in multiple rounds, jointly with the decision functions. In either case, the problem could then be formulated as a reinforcement learning (RL) problem [73], which has been successfully applied to many communication problems [74].

In a *naive* approach to solve optimization problem (9), each agent assumes that its information about the world is perfect, and all the other agents share the same information. Hence, the optimization problem solved by agent j is

$$\left(\ldots,\boldsymbol{s}_j^{\text{naive}},\ldots\right)=\underset{\boldsymbol{s}_1,\ldots,\boldsymbol{s}_n}{\operatorname{argmax}}\,\mathrm{E}\left[u(\mathbf{y}_j,\boldsymbol{s}_1(\mathbf{y}_j),\ldots,\boldsymbol{s}_n(\mathbf{y}_j))\right]. \quad (10)$$

This approach can be improved by taking into account the imperfection in $\mathbf{y}_j$ with respect to $\mathbf{x}$, i.e., taking the expectation over $p_{\mathbf{x}\mathbf{y}_j}$ as conventionally done in robust signal processing [75], instead of simply taking $\mathbf{y}_j$ as being perfect, i.e., $\mathbf{y}_j=\mathbf{x}$. Yet, it is still fundamentally limited as the decentralized information structure is not taken into account: Coordination cannot be reached. The alternative *best-response* strategy is optimal given the strategies of the other agents, i.e., a *Nash equilibrium* [76]. Hence, best-response strategies $(\boldsymbol{s}_1^{\mathrm{BR}},\ldots,\boldsymbol{s}_n^{\mathrm{BR}})$ satisfy:

$$\boldsymbol{s}_j^{\mathrm{BR}}=\underset{\boldsymbol{s}_j}{\operatorname{argmax}}\,\mathrm{E}\left[u(\mathbf{x},\boldsymbol{s}_j,\bar{\boldsymbol{s}}_j^{\mathrm{BR}})\right], \quad (11)$$

where we have used $\bar{\boldsymbol{s}}_j^{\mathrm{BR}}$ as a short-hand notation for all strategies $\boldsymbol{s}_k^{\mathrm{BR}}$ except $k=j$, and omitted the functional dependencies for the sake of clarity. A best-response strategy is also called a *per-agent optimal strategy*, and can be reached by iterating over the agents, which transforms the decentralized optimization problem (9) into a succession of conventional centralized functional optimization problems that can be tackled with conventional optimization tools. Yet, this best-response solution suffers from two important limitations. First, it only enables a *weak form of cooperation* as solutions necessitating a tight inter-dependency between the agents cannot be reached. More specifically, each agent can only update its action *unilaterally*. This means that a solution necessitating several agents to update their strategies at the same time *cannot* be reached. Second, it still requires solving a functional optimization problem at each agent, and hence, is severely limited by the complexity when the dimension of the problem grows.

1) Centralized Training of Decentralized Strategies: We will now discuss how the Team-DNN (T-DNN) approach proposed in [77] allows to leverage recent developments in deep learning to solve the two main challenges: (i) achieving a strong form of cooperation, and (ii) reducing the complexity. This approach is extended in [78] to the design via DNN of instantaneous and quantized message exchanges between the transmitters, where it is highlighted how joint optimization of message sharing and transmission provide more robustness. It is further extended to other settings and arbitrary constraints in [79].

As a first step, optimizing (9) over the space of functions, it is natural to resort to a set of basis functions to reduce the dimensionality of the optimization space (see e.g., [80]). Hence, we propose to restrict the strategy of agent j to belong to a parameterized subspace, i.e., to be of the form $\boldsymbol{s}_j^{\boldsymbol{\theta}_j}$, $\boldsymbol{\theta}_j$ being a vector of real parameters. We will consider DNNs to parametrize the decision functions for their many advantages; in particular, for their efficient implementation and the abundant literature [1], [81], but other functional approximation methods could also be considered. Optimization problem (9) can be approximated as:

$$(\boldsymbol{\theta}_1^\star,\ldots,\boldsymbol{\theta}_n^\star)=\underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n}{\operatorname{argmax}}\,\mathrm{E}\left[u(\mathbf{x},\boldsymbol{s}_1^{\boldsymbol{\theta}_1}(\mathbf{y}_1),\ldots,\boldsymbol{s}_n^{\boldsymbol{\theta}_n}(\mathbf{y}_n))\right]. \quad (12)$$

Following a data-driven approach, we then aim at maximizing the average performance using the training samples from the known distribution. This is possible as the objective utility function u is known and differentiable. This will be achieved by *centralized training* to optimize over *decision functions* using the training samples. In practice, this means that we will jointly update the parameter vectors of all the agents $(\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n)$ using the stochastic gradient approach during the training phase, as is standard in deep learning, i.e., at step k

$$\begin{aligned}(\boldsymbol{\theta}_1^{(k)},\ldots,\boldsymbol{\theta}_n^{(k)})&=(\boldsymbol{\theta}_1^{(k-1)},\ldots,\boldsymbol{\theta}_n^{(k-1)})\\&\quad+\alpha_k\nabla_{(\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n)}u(\mathbf{x},\boldsymbol{s}_1^{\boldsymbol{\theta}_1^{(k-1)}}(\mathbf{y}_1),\ldots,\boldsymbol{s}_n^{\boldsymbol{\theta}_n^{(k-1)}}(\mathbf{y}_n)). \quad (13)\end{aligned}$$

We illustrate the proposed T-DNN approach in Fig. 4. Interestingly, it remains an open problem to determine how

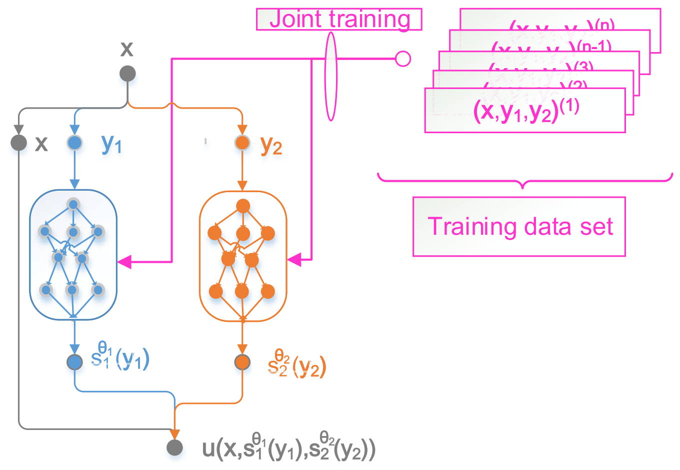

Fig. 4. Illustration of the T-DNN approach with centralized training and decentralized testing.

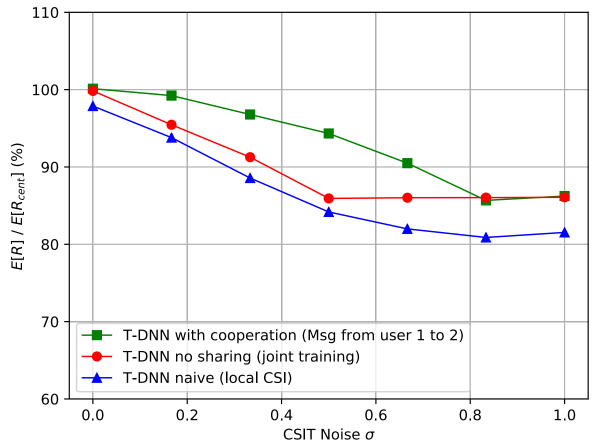

Fig. 5. Percentage of the average sum-rate achieved by a centralized DNN as a function of the maximum transmit power P for different precoding schemes.

efficient these methods designed in the *centralized setting* work in the decentralized setting at hand. In particular, it is not known which DNN architectures are better suited for the decentralized setting and how to improve the efficiency of the training.

2) Application in Wireless Networks: Learning to Cooperate in Coordinated Power Control: To illustrate the application of the concepts described above in wireless networks, we consider a toy example consisting of 2 single-antenna transmitters, with imperfect estimates of the channel coefficients, serving 2 single-antenna receivers. We consider a standard Rayleigh fading scenario and joint precoding across the transmitters with the goal of maximizing the sum rate. We also extend the previous team decision formulation (9) to allow *a one stage* limited exchange of information between the two transmitters.

We consider a T-DNN architecture, where a different DNN is used to parameterize each decision function, while all the DNNs are trained jointly. Following the assumption of one-step exchange, the two DNNs generate the messages to be exchanged, while they also learn the power control from all the inputs.

We consider a simple CSI configuration where transmitter 1 has a noisy CSI, where all the coefficients are corrupted by an additive independent Gaussian noise of variance σ^2, while transmitter 2 has access to perfect CSI. To facilitate the qualitative interpretation, we furthermore reduce to an asymmetric setting where only transmitter 1 can share a message with transmitter 2 via one-step cooperation.

In Fig. 5, we show the average sum rate after normalization by the performance achieved if perfect CSI is handed over to a central node controlling both transmitters, which serves as an upperbound on the performance. We first observe that the *naive* use of DNNs in which each transmitter applies its learning algorithm assuming that it controls the two transmit antennas (i.e., transmitter j is trained using only samples of the locally available CSI) is outperformed by the proposed T-DNN approach, which is hence more robust to the distributed CSI configuration at hand. We can also notice the benefit of a cooperation link: The T-DNNs have learned during training how to use the limited cooperation link between them to exchange useful information; and hence, to coordinate in order to maximize the sum rate.

VI. Learning at the Wireless Edge

There is an ongoing rapid growth in IoT applications, which depend heavily on data collected by sensor nodes being continuously communicated to centralized processing units, typically located at the network edge, made possible by the emerging multi-access edge computing (MEC) paradigm. Data collected at these centralized units is processed to make inferences and prediction on the state of the system being monitored, which in turn may lead to status updates that are communicated to users, or action instructions delivered to actuators.

ML tools are increasingly deployed for the analysis of huge amount of data collected from IoT devices. With an increasing number of successful and promising IoT applications and deployments, we expect that communication of IoT data for learning tasks will constitute a significant portion of the wireless network traffic in the near future. However, there are two potential roadblocks in front of this MEC-based centralized training approach. First of all, offloading all the data to a cloud processor for centralized training will be challenging particularly in wireless networks with limited bandwidth and energy resources. This is particularly true for data intensive applications, such as autonomous vehicles or virtual reality. For example, a self-driving car is expected to generate about one gigabyte of data per second, and continuously offloading such an amount of data to the edge network is not realistic. Privacy is another concern that can prevent centralized ML for most sensor data collected by IoT devices, e.g., smart meters [82] or electric vehicles [83]. While local processing of IoT data is an alternative, often a single device is limited in terms of both available data and computation power. An alternative is to implement learning at the wireless edge, also called *edge learning* [84], in the form of distributed stochastic gradient descent (DSGD) or *federated learning (FL)* [85].

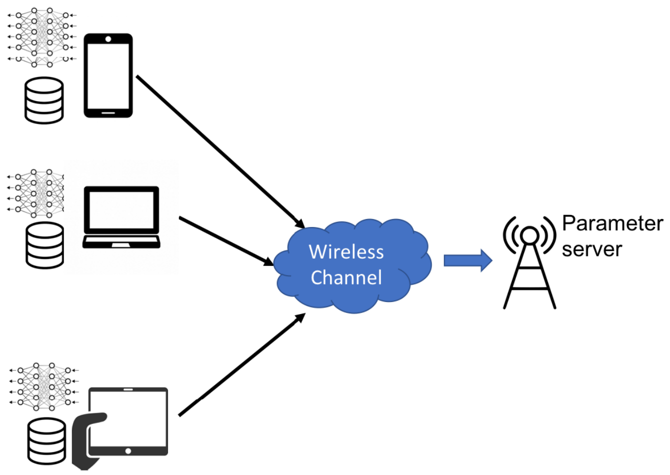

Fig. 6. Wireless edge learning where various devices communicate their local model estimates/ gradients to the parameter server over a shared wireless channel.

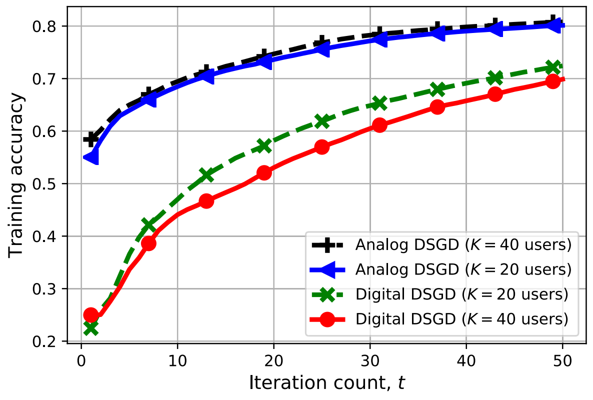

Fig. 7. Accuracy of analog and digital transmission schemes for wireless edge learning with different number of users.

It is now commonly accepted that the main bottleneck in distributed learning is the communication load [86]. Due to the lack of centralized processing capabilities, these algorithms depend heavily on information exchange between multiple learning agents, representing different devices each with its own local dataset, either through a 'master' orchestrating node, called a *parameter server*, or in a fully distributed fashion through device-to-device communications. In either case, distributed learning requires iterative information exchanges among the participating devices and the parameter server, where the devices share either their local gradient estimates in DSGD, or local model updates in FL.

There have been numerous studies that focus on communication-efficient distributed ML. These studies can be grouped into three different approaches, namely *quantization*, *sparsification*, and *local updates*. Quantization algorithms aim at reducing the amount of information that need to be communicated to convey the result of local learning iteration, e.g., the local gradient estimate [87], [88]. Sparsification, on the other hand, reduces the communication load by transmitting only the important values of local estimates [89]–[91]. Another approach is to reduce the frequency of communication from the devices by allowing local parameter updates [92], [93]. We remark, however, that, these studies do not explicitly model the underlying communication channel between the devices and the parameter server, and mainly focus on large scale distributed learning within server farms, where hundreds, maybe thousands of machines collaborate to learn a high-dimensional model on an extremely large dataset. However, as we will show below, taking the particular channel model into account is critical in wireless edge learning, where the channel can be severely limiting.

Consider DSGD over a shared wireless medium, as illustrated in Fig. 6, where the transmission of local gradient estimates from the devices participating in the learning process to the parameter server can be formulated as a wireless computation problem [94]. One approach to this problem is to treat communication and computation separately, and exploit a coding scheme across computing agents such that each of them is assigned a non-zero rate to convey its gradient estimate at each iteration. Therefore, each agent employs quantization to reduce the amount of information to be transmitted to the level that is allowed by the wireless channel. This can be called a 'separate digital' scheme as the gradient estimates are converted into bits, which are communicated by independent channel codes.

Note, however, that, the parameter server is interested only in the average of the gradient estimates, rather than their individual values. Accordingly, a much more efficient communication strategy would be to transmit local estimates without any coding, in an 'analog' fashion. If the devices are synchronized, than the wireless channel adds their estimates, directly conveying the desired value to the parameter server (which simply divides this sum by the number of devices to find the average). A random projection of the gradient estimates is proposed in [94] to reduce the required channel bandwidth. This approach can also be extended to the scenario with fading [95], [96], in which case power control can be employed at the devices to align their transmissions at the same received power level.

In Fig. 7 we illustrate the performance of the digital and analog computation approaches for learning over the wireless edge. The figure compares the training accuracy when a single layer NN is trained on the MNIST dataset. A total of 60000 data samples are distributed across K devices, which employ DSGD utilizing ADAM optimizer. The figure compares the accuracy achieved for a fixed average transmit power value for each user. We observe that analog transmission of gradient estimates achieves a significantly higher accuracy compared to first quantizing the estimates, and then transmitting the quantized bits with a channel code. We also make an interesting observation from Fig. 7: while the accuracy of the analog scheme increases with the number of devices, as each additional device comes with its own power source, the digital scheme has an optimal number of devices, beyond which the accuracy degrades. This is because, channel resources per device becomes limited beyond this optimal number of devices, which, in turn, limits the accuracy of the gradient estimates that are conveyed to the parameter server.

Overall, the results highlight the fact that, for efficient ML at the wireless edge, communication and computation have to be considered jointly, and distributed ML can benefit from physical layer techniques to improve the efficiency and accuracy. A similar observation is also made in [97] by considering coded wireless computation in the map-shuffle-reduce framework, where physical layer techniques are leveraged to provide robustness against both device and channel uncertainties.

VII. Conclusions

We have presented what we hope is a stimulating overview of the promises and challenges of ML for the physical layer of wireless networks. We have presented a wide variety of wireless communications problems, in which ML tools have been shown to offer significant gains. As indicated earlier, these correspond to scenarios in which either we do not have an accurate model of the system, or we have an accurate model but the optimal solution is extremely complex and thus cannot be attained with conventional means. Various power allocation problems have been presented as good examples of the latter scenario. The joint source-channel coding problem can be considered as exhibiting both limitations. In the case of image transmission, we do not have a good statistical model of natural images; however, even when transmitting Gaussian sources over a noisy channel, the optimal solution is not known for finite blocklengths, as the separation theorem fails. We have also highlighted another connection between ML and the wireless physical layer through edge learning. We have shown that the accuracy of distributed ML over wireless channels can benefit greatly from the joint treatment of the physical layer and the employed learning algorithm.

In light of these intriguing achievements and challenges, an important question that our research community will tackle in the next few years is the following: Will the ongoing ML revolution completely transform communication system design, so that we will soon be designing *autonomous communication devices* that do not need standards or protocols, and can simply learn to communicate with one another using data-driven ML techniques? Or do the existing drawbacks of ML-based techniques limit their relevance for communication systems, and we should instead "stick to our guns" – the time-tested highly-optimized model-based approaches? While time will tell what the answer is, it will probably land somewhere in the middle; strong domain knowledge and model-based approaches will need to be combined with powerful data-driven ML techniques. Another important question relates to the use of physical layer techniques for edge learning: Given the rapid speed of developments in ML and particularly edge learning, do we need new standards and new communication techniques that can sustain the growing demand for ML applications at the edge?

We believe that ML and data-driven approaches in general have a lot to offer to all aspects of the communication network architecture, and they have already started to have impact on the higher layers [98]–[100]. Yet, to realize this promise, significant research efforts are needed, from adaptation of existing ML techniques to the development of new ones that can meet the constraints and requirements of communication networks, including the implementation of at least some of these capabilities in low-power chips that can be used in mobile devices [101], [102], and/ or the development of fully distributed, yet efficient implementations that can employ low-power low-complexity mobile devices.

To conclude, one message that comes out loud and clear from our recent experience with deep learning and other data-driven approaches is that we should think big and be bold. Communications engineers are trained to think about physical models and optimal solutions, but the success of deep learning hinges on using lots of data together with 'naive' lightweight approaches, like stochastic gradient descent, to solve NP-hard problems. It takes quite a bit of cultural transformation to digest this. Moreover, the performance of these generic lightweight tools can be improved significantly through domain expertise in wireless communications, complemented with a thorough knowledge of the tricks of the trade in ML.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[3] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[4] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA, USA: Kluwer, 1991.

[5] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory*, vol. IT-23, no. 3, pp. 337–343, May 1977.

[6] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inf. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

[7] *System Architecture for the 5G System, LTE Release 15*, document 3GPP TS 23.501, v 15.1.0, Mar. 2018.

[8] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 3rd Quart., 2019. doi: 10.1109/COMST.2019.2904897.

[9] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Machine learning for wireless networks with artificial intelligence: A tutorial on neural networks," Oct. 2017, *arXiv:1710.02913*. [Online]. Available: http://arxiv.org/abs/1710.02913

[10] L. Liang, H. Ye, and G. Y. Li, "Toward intelligent vehicular networks: A machine learning framework," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 124–135, Feb. 2019.

[11] J. Jagannath, N. Polosky, A. Jagannath, F. Restuccia, and T. Melodia, "Machine learning for wireless communications in the Internet of Things: A comprehensive survey," *Ad Hoc Netw.*, vol. 93, 2019, Art. no. 101913.

[12] X. Zhou, M. Sun, G. Y. Li, and B. F. Juang, "Intelligent wireless communications enabled by cognitive radio and machine learning," *China Commun.*, vol. 15, no. 12, pp. 16–48, Dec. 2018.

[13] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 4, pp. 648–664, Dec. 2018.

[14] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," Feb. 2017, *arXiv:1702.08608*. [Online]. Available: https://arxiv.org/abs/1702.08608

[15] K. Holstein, J. W. Vaughan, H. Daumé, III, M. Dudík, and H. Wallach, "Improving fairness in machine learning systems: What do industry practitioners need?" in *Proc. CHI Conf. Hum. Factors Comput. Syst.* New York, NY, USA: ACM, Scotland, U.K., 2019, pp. 600-1–600-16.

[16] T. O'Shea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proc. GNU Radio Conf.*, vol. 1, 2016, no. 1. [Online]. Available: https://pubs.gnuradio.org/index.php/grcon/article/view/11

[17] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2019, pp. 1–8.

[18] I. Nascimento, F. Mendes, M. Dias, A. Silva, and A. Klautau, "Deep learning in rat and modulation classification with a new radio signals dataset," in *Proc. 36th Simposio Brasileiro Telecomunicacoes Processamento Sinais (SBrT)*, Campina Grande, Brazil, Sep. 2018.

[19] M. Arnold, J. Hoydis, and S. ten Brink, "Novel massive MIMO channel sounding data applied to deep learning-based indoor positioning," in *Proc. Int. ITG Conf. Syst., Commun. Coding (SCC)*, Feb. 2019, pp. 1–6.

[20] 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, Oct. 2018.

[21] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2015, pp. 2692–2700. [Online]. Available: http://papers.nips.cc/paper/5866-pointer-networks.pdf

[22] A. Milan, S. Rezatofighi, R. Garg, A. Dick, and I. Reid, "Data-driven approximations to NP-hard problems," in *Proc. AAAI Conf. Artif. Intell.*, 2017. [Online]. Available: https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14700

[23] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," in *Proc. NIPS Deep Learn. Workshop*, 2013.

[24] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[25] N. Farsad and A. Goldsmith, "Neural network detection of data sequences in communication systems," *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5663–5678, Nov. 2018.

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

[27] 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, Feb. 2018.

[28] P. Jiang *et al.*, "Artificial intelligence-aided OFDM receiver: Design and experimental results," Dec. 2018, *arXiv:1812.06638*. [Online]. Available: https://arxiv.org/abs/1812.06638

[29] E. Nachmani Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2016, pp. 341–346.

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

[31] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," in *Proc. Conf. Inf. Sci. Syst. (CISS)*, 2017, pp. 1–6.

[32] A. Felix, S. Cammerer, S. Dörner, J. Hoydis, and S. Ten Brink, "OFDM-autoencoder for end-to-end learning of communications systems," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[33] V. Raj and S. Kalyani, "Backpropagating through the air: Deep learning at physical layer without channel models," *IEEE Commun. Lett.*, vol. 22, no. 11, pp. 2278–2281, Nov. 2018.

[34] D. Neumann, T. Wiese, and W. Utschick, "Learning the MMSE channel estimator," *IEEE Trans. Signal Process.*, vol. 66, no. 11, pp. 2905–2917, Jan. 2018.

[35] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 652–655, Apr. 2019.

[36] M. Koller, C. Hellings, M. Knoedlseder, T. Wiese, D. Neumann, and W. Utschick, "Machine learning for channel estimation from compressed measurements," in *Proc. 15th Int. Symp. Wireless Commun. Syst. (ISWCS)*, 2018, pp. 1–5.

[37] H. He, C. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmwave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, Oct. 2018.

[38] D. J. Love, R. W. Heath, Jr., V. K. Lau, D. Gesbert, B. D. Rao, and M. Andrews, "An overview of limited feedback in wireless communication systems," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 8, pp. 1341–1365, Oct. 2008.

[39] P. Kuo, H. T. Kung, and P. Ting, "Compressive sensing based channel feedback protocols for spatially-correlated massive antenna arrays," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 492–497.

[40] X. Rao and V. K. N. Lau, "Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3261–3271, Jun. 2014.

[41] R. Setiono and G. Lu, "Image compression using a feedforward neural network," in *Proc. IEEE Int. Conf. Neural Netw., IEEE World Congr. Comput. Intell.*, vol. 7, Jun. 1994, pp. 4761–4765.

[42] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Apr. 2017, pp. 1–27.

[43] J. Han, S. Lombardo, C. Schroers, and S. Mandt, "Deep probabilistic video compression," *CoRR*, vol. abs/1810.02845, Oct. 2018. [Online]. Available: https://arxiv.org/abs/1810.02845

[44] S. Kankanahalli, "End-to-end optimized speech coding with deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2521–2525.

[45] *Spatial Channel Model for Multiple Input Multiple Output (MIMO) Simulations*, document, 3GPP 3GPP2 Spatial Channel Model Ad-hoc Group3GPP, Sep. 2003.

[46] *IST-4-027756 WINNER II D1.1.2 V1.2 WINNER II Channel Models*. Accessed: Jan. 25, 2019. [Online]. Available: https://www.cept.org/files/8339/winner2%20-%20final%20report.pdf

[47] V. Erceg *et al.*, *Channel Models for Fixed Wireless Applications*, Standard IEEE 802.16.3c-01/29r4, IEEE 802.16 Broadband Wireless Access Working Group, 2001.

[48] L. Liu *et al.*, "The COST 2100 MIMO channel model," *IEEE Wireless Commun.*, vol. 19, no. 6, pp. 92–99, Dec. 2012.

[49] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 416–419, Apr. 2019.

[50] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang, "MIMO channel information feedback using deep recurrent network," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 188–191, Jan. 2019.

[51] Z. Liu, L. Zhang, and Z. Ding, "Exploiting bi-directional channel reciprocity in deep learning for low rate massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. .8, no. 3, pp. 889–892, Jun. 2019.

[52] L. Theis, W. Shi, A. Cunnigham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[53] G. Toderici *et al.*, "Variable rate image compression with recurrent neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016.

[54] Q. Yang, M. B. Mashhadi, and D. Gündüz, "Deep convolutional compression for massive MIMO CSI feedback," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, 2019.

[55] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Dec. 2016, pp. 223–228.

[56] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[57] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–5.

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

[59] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," *IEEE Trans. Cogn. Commun. Netw.*, to be published.

[60] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, Jul. 2006.

[61] D. B. Kurka and D. Gündüz, "Successive refinement of images with deep joint source-channel coding," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2019.

[62] M. S. Ibrahim, A. S. Zamzam, X. Fu, and N. D. Sidiropoulos, "Learning-based antenna selection for multicasting," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jun. 2018, pp. 1–5.

[63] V. Ntranos, N. D. Sidiropoulos, and L. Tassiulas, "On multicast beamforming for minimum outage," *IEEE Trans. Wireless Commun.*, vol. 8, no. 6, pp. 3172–3181, Jun. 2009.

[64] Y. Shi, A. Konar, N. D. Sidiropoulos, X. Mao, and Y. Liu, "Learning to beamform for minimum outage," *IEEE Trans. Signal Process.*, vol. 66, no. 19, pp. 5180–5193, Oct. 2018.

[65] S. S. Christensen, R. Agarwal, E. De Carvalho, and J. M. Cioffi, "Weighted sum-rate maximization using weighted MMSE for MIMO-BC beamforming design," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4792–4799, Dec. 2008.

[66] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "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, Sep. 2011.

[67] J. Kaleva, A. Tolli, and M. Juntti, "Successive convex approximation for simultaneous linear TX/RX design in MIMO BC," in *Proc. IEEE Asilomar Conf. Signals, Syst. Comput. (ACSSC)*, Nov. 2015, pp. 1227–1231.

[68] 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," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sapporo, Japan, Jul. 2017, pp. 1–6.

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

[70] R. Radner, "Team decision problems," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 857–881, 1962.

[71] J. Marschak and R. Radner, *Economic Theory of Teams*. London, U.K.: Yale Univ. Press, Feb. 1972.

[72] D. Gesbert and P. de Kerret, "Team methods for device cooperation in wireless networks," in *Cooperative and Graph Signal Processing*, P. M. Djuric and C. Richard, Eds. New York, NY, USA: Academic, 2018, ch. 18, pp. 469–487.

[73] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[74] N. C. Luong *et al.*, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tuts.*, to be published.

[75] D. A. Awan, R. L. G. Cavalcante, and S. Stanczak, "A robust machine learning method for cell-load approximation in wireless networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2601–2605.

[76] J. Nash, "Non-cooperative games," *Ann. Math.*, vol. 54, no. 2, pp. 286–295, 1951.

[77] P. de Kerret and D. Gesbert, "Robust decentralized joint precoding using team deep neural network," in *Proc. IEEE Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2018, pp. 1–5.

[78] M. Kim, P. de Kerret, and D. Gesbert, "Learning to cooperate in decentralized wireless networks," in *Proc. 52nd Asilomar Conf. Signals, Syst. Comput.*, Pacific Grove, CA, USA, Oct. 2018.

[79] H. Lee, S. Hyun, and T. Q. S. Quek, "Deep learning for distributed optimization: Applications to wireless resource management," *IEEE J. Sel. Areas Commun.*, to be published.

[80] 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*, J. van Leeuwen, A. Muscholl, D. Peleg, J. Pokorný, and B. Rumpe, Eds. Berlin, Germany: Springer, 2010, pp. 440–451.

[81] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade* (Lecture Notes in Computer Science), G. Montavon, G. B. Orr, and K.-R. Müller, Eds., 2nd ed. Berlin, Germany: Springer, 2012, pp. 9–48.

[82] G. Giaconi, D. Gunduz, and H. V. Poor, "Privacy-aware smart metering: Progress and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 6, pp. 59–78, Nov. 2018.

[83] N. Saxena, S. Grijalva, V. Chukwuka, and A. V. Vasilakos, "Network security and privacy challenges in smart vehicle-to-grid," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 88–98, Aug. 2017.

[84] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," 2018, *arXiv:1812.02858*. [Online]. Available: http://arxiv.org/abs/1812.02858

[85] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527

[86] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Lane Red Hook, NY, USA: Curran Associates, 2014, pp. 19–27.

[87] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proc. ICML*, Jul. 2015, pp. 1737–1746.

[88] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs," in *Proc. INTERSPEECH*, Singapore, Sep. 2014, pp. 1058–1062.

[89] N. Strom, "Scalable distributed DNN training using commodity GPU cloud computing," in *Proc. INTERSPEECH*, 2015, pp. 1488–1492.

[90] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," Jul. 2017, *arXiv:1704.05021v2*. [Online]. Available: http://arxiv.org/abs/1704.05021v2

[91] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Sparse binary compression: Towards distributed deep learning with minimal communication," 2018, *arXiv:1805.08768*. [Online]. Available: http://arxiv.org/abs/1805.08768

[92] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, A. Singh and J. Zhu, Eds. Fort Lauderdale, FL, USA: PMLR, Apr. 2017, pp. 1273–1282.

[93] T. Chen, G. B. Giannakis, T. Sun, and W. Yin, "LAG: Lazily aggregated gradient for communication-efficient distributed learning," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst. (NIPS)*, USA, 2018, pp. 5055–5065.

[94] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," 2019, *arXiv:1901.00844*. [Online]. Available: http://arxiv.org/abs/1901.00844

[95] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," 2019, *arXiv:1907.09769*. [Online]. Available: https://arxiv.org/abs/1907.09769

[96] G. Zhu, Y. Wang, and K. Huang, "Low-latency broadband analog aggregation for federated edge learning," *CoRR*, vol. abs/1812.11494, 2018.

[97] S. Ha, J. Zhang, O. Simeone, and J. Kang, "Coded federated computing in wireless networks with straggling devices and imperfect CSI," 2019, *arXiv:1901.05239*. [Online]. Available: http://arxiv.org/abs/1901.05239

[98] R. Li *et al.*, "Intelligent 5G: When cellular networks meet artificial intelligence," *IEEE Wireless Commun.*, vol. 24, no. 5, pp. 175–183, Oct. 2017.

[99] M. G. Kibria, K. Nguyen, G. P. Villardi, O. Zhao, K. Ishizu, and F. Kojima, "Big data analytics and artificial intelligence in next-generation wireless networks," *IEEE Access*, vol. 6, pp. 32328–32338, 2018.

[100] V. P. Kafle, Y. Fukushima, P. Martinez-Julia, and T. Miyazawa, "Consideration on automation of 5G network slicing with machine learning," in *Proc. ITU Kaleidoscope, Mach. Learn. 5G Future (ITU K)*, Nov. 2018, pp. 1–8.

[101] S. I. Venieris, A. Kouris, and C. Bouganis, "Deploying deep neural networks in the embedded space," 2018, *arXiv:1806.08616*. [Online]. Available: http://arxiv.org/abs/1806.08616

[102] H. Yoo, "Intelligence on silicon: From deep-neural-network accelerators to brain mimicking AI-SoCs," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 20–26.

Deniz Gündüz (S'03–M'08–SM'13) received the B.S. degree in electrical and electronics engineering from METU, Turkey, in 2002, and the M.S. and Ph.D. degrees in electrical engineering from NYU Tandon School of Engineering (formerly Polytechnic University) in 2004 and 2007, respectively. After his Ph.D., he served as a Post-Doctoral Research Associate with Princeton University, and as a Consulting Assistant Professor with Stanford University. He was a Research Associate with CTTC, Barcelona, Spain, until September 2012, when he joined the Electrical and Electronic Engineering Department, Imperial College London, U.K., where he is currently a Reader (Associate Professor) in information theory and communications, and leads the Information Processing and Communications Laboratory (IPC-Lab).

His research interests lie in the areas of communications and information theory, machine learning, and privacy. He was a recipient of the IEEE Communications Society—Communication Theory Technical Committee (CTTC) Early Achievement Award in 2017, a Starting Grant of the European Research Council (ERC) in 2016, IEEE Communications Society Best Young Researcher Award for the Europe, Middle East, and Africa Region in 2014, Best Paper Award at the 2016 IEEE Wireless Communications and Networking Conference (WCNC), and the Best Student Paper Awards at the 2018 IEEE Wireless Communications and Networking Conference (WCNC), and the 2007 IEEE International Symposium on Information Theory (ISIT). He was the General Co-Chair of the 2019 London Symposium on Information Theory, 2018 International ITG Workshop on Smart Antennas, 2016 IEEE Information Theory Workshop, and 2012 European School of Information Theory. He is an Editor of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, and a Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, Special Issue on Machine Learning in Wireless Communication. He served as an Editor of the TRANSACTIONS ON COMMUNICATIONS from 2013 to 2018.

Paul de Kerret received the degree in engineering from the French Graduate School IMT Atlantique in 2010 and the Diploma degree in electrical engineering and information technology from the Munich University of Technology through a double degree program, and the Ph.D. degree from the French Graduate School Télécom Paris in 2013. Since 2015, he has been a Senior Researcher with EURECOM as part of the ERC-funded project PERFUME to investigate how to enable efficient and decentralized cooperation to boost performance in future wireless networks. He is particularly active in the decentralized use of machine learning methods to solve coordination problems. He has been involved in several European collaborative projects on mobile communications, co-presented several tutorials at major IEEE international conferences, and authored more than 30 papers in IEEE flagship conferences.

Nicholas D. Sidiropoulos (F'09) received the Diploma degree in electrical engineering from the Aristotelian University of Thessaloniki, Thessaloniki, Greece, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland at College Park, College Park, MD, USA, in 1988, 1990, and 1992, respectively. He has served on the Faculty of the University of Virginia (UVA), University of Minnesota, and the Technical University of Crete, Greece, prior to his current appointment as a Louis T. Rader Professor and a Chair of the Electrical and Computer Engineering Department, UVA.

His research interests are in signal processing, communications, optimization, tensor decomposition, and factor analysis, with applications in machine learning and communications. He received the NSF/CAREER award in 1998, the IEEE Signal Processing Society (SPS) Best Paper Award in 2001, 2007, and 2011, served as IEEE SPS Distinguished Lecturer (2008–2009), and currently serves as Vice President—Membership of IEEE SPS. He received the 2010 IEEE Signal Processing Society Meritorious Service Award, and the 2013 Distinguished Alumni Award from the University of Maryland, Dept. of ECE. He is a fellow of EURASIP in 2014.

David Gesbert (S'96–M'99–SM'06–F'11) received the Ph.D. degree from the Ecole Nationale Superieure des Telecommunications, France, in 1997. From 1997 to 1999, he was with the Information Systems Laboratory, Stanford University. He was a Founding Engineer of Iospan Wireless, Inc., a Stanford spin off pioneering MIMO-OFDM (now Intel). Before joining EURECOM in 2004, he has been with the Department of Informatics, University of Oslo, as an Adjunct Professor. He is currently a Professor and the Head of the Communication Systems Department, EURECOM. He has published about 300 papers and 25 patents, some of them winning 2019 ICC Best Paper Award, 2015 IEEE Best Tutorial Paper Award (Communications Society), 2012 SPS Signal Processing Magazine Best Paper Award, 2004 IEEE Best Tutorial Paper Award (Communications Society), 2005 Young Author Best Paper Award for Signal Proc. Society journals, and paper awards at conferences 2011 IEEE SPAWC, 2004 ACM MSWiM. He has been a Technical Program Co-Chair for ICC2017. He was named a Thomson-Reuters Highly Cited Researchers in Computer Science. Since 2015, he holds the ERC Advanced grant PERFUME on the topic of smart device Communications in future wireless networks. Since 2019, he has been the Head of the Huawei-funded Chair on Advanced Wireless Systems Towards 6G Networks. Since 2017, he has also been a Visiting Academic Master within the Program 111 at the Beijing University of Posts and Telecommunications and a member in the Joint BUPT-EURECOM Open5G Lab. He is a Board Member for the OpenAirInterface (OAI) Software Alliance.

Chandra R. Murthy received the B.Tech. degree in electrical engineering from IIT Madras, Chennai, India, in 1998, the M.S. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2000, and Ph.D. degree in electrical and computer engineering from the University of California at San Diego, San Diego, CA, USA, in 2006. From 2000 to 2002, he was an Engineer with Qualcomm, Inc., San Jose, USA, where he worked on WCDMA baseband transceiver design and 802.11b baseband receivers. From 2006 to 2007, he was a Staff Engineer with Beceem Communications, Inc., Bengaluru, India, on advanced receiver architectures for the 802.16e Mobile WiMAX standard. He is currently a Professor with the Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru.

His research interests are in the areas of energy harvesting communications, multiuser MIMO systems, and sparse signal recovery techniques applied to wireless communications. He is an Elected Member of the IEEE SPCOM Technical Committee from 2014 to 2016, and has been re-elected from 2017 to 2019 term. His paper received the Best Paper Award in the Communications Track at NCC 2014 and a paper coauthored with his student received the Student Best Paper Award at the IEEE ICASSP 2018. He has more than 50 journal papers and more than 80 conference papers to his credit. He was an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS from 2012 to 2016. He is a Past Chair of the IEEE Signal Processing Society, Bangalore Chapter. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING and the IEEE TRANSACTIONS ON INFORMATION THEORY, and as an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS.

Mihaela van der Schaar (F'09) is currently a John Humphrey Plummer Professor of machine learning, artificial intelligence and medicine with the University of Cambridge and also a Turing Fellow with The Alan Turing Institute, London, where she leads the effort on data science and machine learning for personalised medicine. She has received the Oon Prize on Preventative Medicine from the University of Cambridge (2018). She has also been a recipient of the NSF Career Award, three IBM Faculty Awards, the IBM Exploratory Stream Analytics Innovation Award, the Philips Make a Difference Award and several best paper awards, including the IEEE Darlington Award. She holds 35 granted USA patents.