Over-the-Air Edge Inference via End-to-End Metasurfaces-Integrated Artificial Neural Networks

Kyriakos Stylianopoulos, *Graduate Student Member, IEEE,* Paolo Di Lorenzo, *Senior Member, IEEE,* and George C. Alexandropoulos, *Senior Member, IEEE*

***Abstract*—In the Edge Inference (EI) paradigm, where a Deep Neural Network (DNN) is split across the transceivers to wirelessly communicate goal-defined features in solving a computational task, the wireless medium has been commonly treated as a source of noise. In this paper, motivated by the emerging technologies of Reconfigurable Intelligent Surfaces (RISs) and Stacked Intelligent Metasurfaces (SIM) that offer programmable propagation of wireless signals, either through controllable reflections or diffractions, we optimize the RIS/SIM-enabled smart wireless environment as a means of over-the-air computing, resembling the operations of DNN layers. We propose a framework of Metasurfaces-Integrated Neural Networks (MINNs) for EI, presenting its modeling, training through a backpropagation variation for fading channels, and deployment aspects. The overall end-to-end DNN architecture is general enough to admit RIS and SIM devices, through controllable reconfiguration before each transmission or fixed configurations after training, while both channel-aware and channel-agnostic transceivers are considered. Our numerical evaluation showcases metasurfaces as instrumental in performing image classification under link budgets that impede conventional communications or metasurface-free systems. It is demonstrated that our MINN framework can significantly simplify EI requirements, achieving near-optimal performance with a 50 dB lower testing signal-to-noise ratio compared to that used in training, even without transceiver channel knowledge.**

***Index Terms*—Edge learning, reconfigurable intelligent surface, stacked intelligent metasurfaces, goal-oriented communications, deep learning, over-the-air computing.**

I. Introduction

In emerging Device-to-Device (D2D) and Internet of Things (IoT) networks, where distributed devices are expected to support various functionalities, such as Integrated Sensing and Communications (ISAC) [1], stringent energy efficiency requirements impose limitations on their communication and computation capabilities. Various sub-networks are envisioned to enable a wide range of high-level applications and services, such as digital twinning through object recognition and computational imaging, or indoor positioning [2], each tied up with a diverse set of requirements to be fulfilled. As a result, it is crucial for the D2D system design to take into account the underlying application. To this end, devising novel communication stacks tailored to the application, which break the traditional network layer taxonomy, comes with reduced implementation overheads.

Goal-Oriented Communications (GOC) [3] signifies a novel communication framework that is gaining popularity in D2D systems, where transmissions of only the necessary information need to take place, instead of the complete data. The goal is implemented by an arbitrary computational function over the data and the Receiver (RX) aims only to perform this computation, rather than reconstruct the input signals. Therefore, GOC reduces messaging overheads and simplifies system architecture [4]. In Edge Inference (EI) approaches, a subset of GOC where the RX wishes to obtain only an estimated label of the data the Transmitter (TX) actually observes and transmits, that split the layers of a Deep artificial Neural Network (DNN) at the endpoints provide significant benefits in terms of communication requirements [5]. According to those methods, low-dimensional feature vectors, that are outputs of intermediate DNN layers, are transmitted over the channel [6]–[8]. The motivation behind such practices is that GOC deviates from the standard Shannon-type communications [9], in that the goal of the system is an arbitrary computational target function over a predetermined distribution of data, rather than objectives derived from the Mutual Information (MI) that call for bit-wise reconstruction of the inputs. In this way, DNNs are employed and trained to capture the joint or conditional data-channel-target distributions.

Current literature has adopted GOC and EI for a wide variety of problems and wireless systems (see [6], [8], [10] and the references therein). However, a common practice across those works is to treat the wireless environment as a source of noise, the effects of which need to be negated at the RX side. The rapid developments of Meta-Surface (MS) technologies for precise Radio-Frequency (RF) domain control open the potential for the wireless propagation medium to be dynamically reconfigurable via reflective Reconfigurable Intelligent Surfaces (RISs) [11] or diffractive Stacked Intelligent Metasurfaces (SIM) [12] with low operational costs. Such MSs have been incorporated in GOC and semantic systems to reduce the transceiver hardware complexity [13] or to enhance the system's rate [14]. Under another research direction, DNN implementations entirely in the RF domain capitalizing on MS-based solutions, have been devised for

This work has been supported by the Smart Networks and Services Joint Undertaking projects TERRAMETA, 6G-DISAC, and 6G-GOALS under the European Union's Horizon Europe research and innovation programme under Grant Agreement numbers 101097101, 101139130, and 101139232 respectively. TERRAMETA also includes top-up funding by UK Research and Innovation under the UK government's Horizon Europe funding guarantee. For this work's experimental results, AWS resources were used which were provided by the National Infrastructures for Research and Technology (GRNET), Greece and funded by the EU Recovery and Resiliency Facility.

K. Stylianopoulos and G. C. Alexandropoulos are with the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 16122 Athens, Greece (e-mails: {kstylianop, alexandg}@di.uoa.gr).

P. Di Lorenzo is with the Department of Information Engineering, Electronics, and Telecommunications, Sapienza University, Italy and CNIT, Italy (e-mail: paolo.dilorenzo@uniroma1.it).

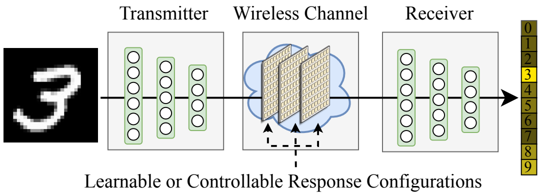

Fig. 1. A Metasurfaces-Integrated artificial Neural Network (MINN) performing Edge Inference (EI) by controlling the wireless propagation channel, which is treated as one or more hidden network layers.

controlled laboratory environments [15]–[17].

In this paper, motivated by the elaborate wireless medium control offered by MS technologies [11], [12], in conjunction with their analog computational capabilities, we design MS-controllable wireless channels that perform Over-the-Air Computation (OAC), under which the comprising metamaterials are treated as hidden artificial neurons that control the wireless medium to perform multi-layer signal processing towards solving EI tasks, as illustrated in Fig. 1. The main contributions of this paper are summarized as follows[1]:

1) We present a novel generic End-to-End (E2E) DNN framework that incorporates MSs-parameterized channels as hidden layers. The framework, entitled Metasurfaces-Integrated Neural Network (MINN), admits variations of controllable MSs or MSs of trainable, yet fixed response configurations, either RISs or SIM, and with or without channel knowledge. We detail the framework's modeling, backpropagation-based training, and deployment from both a theoretical and system architecture perspectives.
2) Different from existing literature that employs MSs at the transceivers, we elaborate on the critical role of reconfiguration of wireless channels as a degree of freedom in optimizing DNN-based EI models that capitalize on programmable OAC, instead of treating the wireless propagation environment as a source of noise. It is showcased that offloading computations onto the channel itself has the potential of greatly reducing the computational costs at the transceivers.
3) We present an extensive numerical evaluation of the proposed MINN framework on an image classification task, which is compared to conventional communication systems, as well as to a baseline in the absence of an MS. The results demonstrate that MS-enabled OAC allows for successful EI with much lower transmit power requirements, even without channel knowledge at the communication ends, since the trained MSs may only rely on static configurations.

The remainder of the paper is structured as follows. Section II details the relevant pieces of theory regarding EI as well as the possible improvements brought by controlling the wireless environment while Section III provides a comprehensive review of the relevant literature. Section IV includes the models used for the considered MS technologies: RIS and SIM, as well as the model for the received signal for both cases. The proposed MINN architecture for EI is presented in Section V, whereas Section VI details its training procedure and discusses the deployment and network considerations. Our numerical investigations are presented in Section VII. Finally, Section VIII includes the paper's concluding remarks.

Notation: Vectors, matrices, and sets are expressed in lowercase bold (e.g., $\mathbf{x}$), uppercase bold (e.g., $\mathbf{X}$), and uppercase calligraphic typefaces (e.g., $\mathcal{X}$ or $\mathcal{X}$), respectively. Apart from $\mathbf{X}^*$, $\mathbf{X}^\dagger$, and $\mathbf{X}^\top$ that denote the conjugate, conjugate transpose, and transpose of $\mathbf{X}$, superscripts and subscripts are used to denote different versions of variables or enumeration over collections of variables depending on context. $[\mathbf{x}]_i$ and $[\mathbf{X}]_{i,j}$ are used to denote the i-th element of $\mathbf{x}$ and the (i,j)-th element of $\mathbf{X}$, respectively. We use variations of the notation $f_{\mathbf{w}}(\cdot)$ to represent neural network functions that are parameterized by their weight matrix $\mathbf{w}$ noting that f may be seen either as a function of its arbitrary input variables (during inference) or as a function of $\mathbf{w}$ (during optimization). $|\mathcal{X}|$ represents the cardinality of the set $\mathcal{X}$, $||\mathbf{X}||_F$ denotes the Frobenius norm of $\mathbf{X}$, $\operatorname{diag}(\mathbf{x})$ creates a square matrix with the elements of $\mathbf{x}$ placed along its main diagonal, and $\operatorname{vec}(\mathbf{X})$ transforms $\mathbf{X}$ into a column vector in a row-by-row fashion. $\otimes$ denotes the Kronecker product, $\{a, b\}$ expresses a set or collection containing a and b, while $\mathbb{1}_{\texttt{cond}}$ is the indicator function equaling to 1 if condition $\texttt{cond}$ holds, otherwise to 0. Finally, $\mathbb{E}_{\mathbf{X}}[\cdot]$ is the expectation operator with respect to the distribution of the random $\mathbf{X}$, and $\jmath \triangleq \sqrt{-1}$.

II. Prerequisites

A. Probabilistic Inference

Given an input observation $\mathbf{x}$, the objective of an inference procedure is to compute and output an associated target attribute value $\mathbf{o} = l(\mathbf{x})$. The mapping function $l(\mathbf{x})$ is considered unknown and intractable to express analytically, therefore, inference involves approximating this relationship through examples of $(\mathbf{x}, \mathbf{o})$ tuples. From a probabilistic perspective, one may fit the conditional Probability Density Function (PDF) of the target $p(\mathbf{o}|\mathbf{x})$ on the available data. In typical settings, point estimates are only required, thus, the problem reduces to predicting the most likely value of $\mathbf{o}$ for a given observation. In the machine learning regime, the previous distribution (or its point estimates) can be approximated by a $\mathbf{w}$-parameterized model $\hat{\mathbf{o}} \triangleq f_{\mathbf{w}}(\mathbf{x})$ that outputs its prediction of the target value $\hat{\mathbf{o}}$ for a given input. Consequently, to solve the inference problem, the parameter values $\mathbf{w}$ need to be optimized. This is achieved by collecting a data set of training tuples $\mathcal{D} \triangleq \{(\mathbf{x}_i, \mathbf{o}_i)\}_{i=1}^{|\mathcal{D}|}$, and then minimizing an amortized cost function over the training instances:

$$J(\mathbf{w}) \triangleq \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \mathfrak{J}(\mathbf{o}_i, \hat{\mathbf{o}}_i), \quad \text{where } \hat{\mathbf{o}}_i = f_{\mathbf{w}}(\mathbf{x}_i). \qquad (1)$$

The per-instance cost function values $\mathfrak{J}(\mathbf{o}_i, \hat{\mathbf{o}}_i)$ quantify the error in the model's predictions and they often have a probabilistic interpretation. For instance, in *classification* settings where each observation belongs to one of d_{cl} classes indexed by the natural number c, the target value is defined via

[1] The code for this paper is available at https://github.com/NoesysLab/Metasurfaces-Integrated-Neural-Networks.

one-hot encoding as $\mathbf{o} = [\mathbb{1}_{c=1}, \mathbb{1}_{c=2}, \ldots, \mathbb{1}_{c=d_{\text{cl}}}]^\top \in \mathbb{N}^{d_{\text{cl}} \times 1}$. The Cross Entropy (CE) loss function is defined as follows:

$$\mathfrak{J}(\mathbf{o}_i, \hat{\mathbf{o}}_i) \equiv J_{\text{CE}}(\mathbf{o}_i, \hat{\mathbf{o}}_i) \triangleq - \sum_{j=1}^{d_{\text{cl}}} [\mathbf{o}_i]_j \log[\hat{\mathbf{o}}_i]_j. \tag{2}$$

Minimizing (2) over $\mathcal{D}$ is equivalent to performing maximum likelihood estimation of $\mathbf{w}$ on $p(\mathbf{o}|\mathbf{x})$ under the assumption that this PDF is a multivariate Bernoulli distribution. Similarly, in *regression* tasks, where $\mathbf{o}, \hat{\mathbf{o}} \in \mathbb{R}^{d_{\text{out}} \times 1}$, the Mean Squared Error (MSE) metric, defined as $1/d_{\text{out}} \|\mathbf{o} - \hat{\mathbf{o}}\|^2$, implies that $p(\mathbf{o}|\mathbf{x})$ is a Gaussian conditional PDF.

B. Artificial Neural Networks

While a wide range of parameterized families of functions is available to use as $f_{\mathbf{w}}(\cdot)$, the current state of the art considers DNN models, capitalizing on their diverse benefits including high expressivity, diverse selection of architectural components for different tasks, and parallelizable computations that offer real-time computational cost during inference on pertinent hardware. Mathematically, let a neural network be expressed as a composition of L layers such that:

$$f_{\mathbf{w}}(\mathbf{x}) \triangleq f^L_{\mathbf{w}^L} \left(f^{L-1}_{\mathbf{w}^{L-1}} \left(\ldots f^1_{\mathbf{w}^1}(\mathbf{x}) \ldots \right) \right), \tag{3}$$

so that the l-th layer ($l = 1, 2, \ldots, L$) is parameterized by $\mathbf{w}_l$, and its output $\bar{\mathbf{o}}^l$ constitutes the input to the $(l+1)$-th layer; this process can be represented recursively by $\bar{\mathbf{o}}^l \triangleq f^l_{\mathbf{w}^l}(\bar{\mathbf{o}}^{(l-1)})$, where we have set $\bar{\mathbf{o}}^0 = \mathbf{x}$. For convenience, let us also define the vector $\mathbf{w} \triangleq [\mathbf{w}_1^\top, \mathbf{w}_2^\top, \ldots, \mathbf{w}_L^\top]^\top$.

We leave aside precise definitions of individual layer functions, as there has been an impressive body of work focusing on layers that perform data-specific computations and capture high-level patterns in data sets [18]. Nevertheless, we highlight the fact that each $f^l_{\mathbf{w}_l}(\cdot)$ is demanded to be a non-linear function, and, in fact, it is often requested to be discriminatory or sigmoidal [19]. These properties guarantee that artificial neural networks with at least two layers (of potentially infinite width) are *universal approximators* and can, therefore, be used to approximate any arbitrary $\mathbf{o} = l(\mathbf{x})$ mapping, i.e., $f_{\mathbf{w}}(\mathbf{x}) \cong l(\mathbf{x})$ [19].

Besides theoretical guarantees, the problem of obtaining $\mathbf{w}$ values that perform successful inference can be efficiently solved by substituting the neural network expression from (3) into (2) (using classification as an example), and subsequently into (1). The latter may be solved through one of the many variations of the Stochastic Gradient Descent (SGD) approach by computing the gradients $\partial J(\mathbf{w})/\partial \mathbf{w}$, and propagating them through the layers of $f_{\mathbf{w}}(\mathbf{x})$ by taking advantage of the chain rule; this leads to the celebrated *backpropagation* algorithm [18] that constitutes the backbone of deep learning.

C. Edge Inference (EI)

The rising field of EI considers the implementation and training of inference tasks over a wireless communication network. In that regard, consider an uplink setup where a multi-antenna TX observes $\mathbf{x}$ and wishes to convey its estimate $\hat{\mathbf{o}}$ of the target value $\mathbf{o}$ to an RX. At first observation, this task is apparently straightforward to implement within the existing frameworks of machine learning and wireless communications under the following two paradigm options:

1) "Infer-then-transmit": The TX may first compute $\hat{\mathbf{o}} = f_{\mathbf{w}}(\mathbf{x})$ and then transmit $\hat{\mathbf{o}}$ over the wireless medium upon performing source coding (i.e., data compression) and channel coding (i.e., modulation and beamforming) to derive the transmission signal $\mathbf{s}$. The above coding computations ensure that a satisfactory communication rate is achievable by the system so that $\hat{\mathbf{o}}$ may be reconstructed at the RX's side via decoding the received signal and decompressing the data. In modern high-complexity wireless systems, these operations require their own optimization procedures and necessitate additional computational costs as well as channel state information knowledge. For most inference tasks, the target values are of a much smaller dimension than the observations. As a result, this option requires small rate requirements, but comes at a computational cost on the TX's side, as the device must be endowed with hardware capable of executing DNN computations locally. Since EI tasks are envisioned specifically for cases where IoT or other lightweight devices with low complexity and minute power consumption send messages to a collection/fusion center, the assumption of computational capabilities for local DNN inference is rather optimistic.

2) "Transmit-then-infer": This converse approach is also possible: the TX performs source and channel coding but no DNN-based EI computations, so that the original observation $\mathbf{x}$ is transmitted instead. In the sequel, the RX performs decoding to obtain the data point, which is then fed to its local $f_{\mathbf{w}}(\cdot)$ to perform inference. While in uplink settings, it is reasonable to assume that the RX has sufficient power and hardware capabilities to support a DNN, the rate required for transmitting the original observation may impose high link budget demands that are not readily met.

A compromise solution to the above two options may be obtainable by exploiting the sequential nature of the DNN structure appearing in (3) [20]. To this end, the following EI option is possible:

3) "Infer-while-transmitting" (DNN splitting): The intermediate representations $\bar{\mathbf{o}}^l$ $\forall l = 1, 2 \ldots, L-1$ can be of arbitrary dimensions, and it is not uncommon to devise architectures with one or more small-sized intermediate layers. In fact, various deep learning models, such as auto-encoders [21] and U-Net [22], are designed specifically to contain such bottleneck layers as a form of compression to keep only relevant information. From that perspective, one may choose to split the first $L' < L$ layers of the DNN to reside at the TX, so that $\bar{\mathbf{o}}^{L'}$ is transmitted over the network and then passed to the $(L'+1)$-th up to the L-th layer in sequence at the RX.

Evidently, the latter paradigm of EI is the most flexible, since one has the option to balance trade-offs between computation and communication resources between the TX and RX. In the remainder of this paper, we will be adopting this paradigm of EI as the default case and will provide further elaboration and extensions, revisiting the two extreme previous cases as baselines in our numerical comparisons.

D. Computational Considerations

When performing DNN splitting over a wireless channel with realistic characteristics (i.e., large- and small-scale fading, as well as Additive White Gaussian Noise (AWGN)), the transmitted output $\bar{\mathbf{o}}^{L'}$ of the L'-th DNN layer will be distorted when it arrives at the RX. By representing the channel state with an abstract random variable $\boldsymbol{\mathcal{H}}$, one needs to minimize the same $\mathbf{w}$-parameterized objective function as before, while accounting for the stochastic nature of the wireless environment (i.e., with respect to $\boldsymbol{\mathcal{H}}$'s distribution), i.e., solve:

$$\mathcal{OP}_{\mathrm{EI}} : \min_{\mathbf{w}} \mathbb{E}_{\boldsymbol{\mathcal{H}}}[J(\mathbf{w})],$$

where each instantaneous channel realization affects the value of the objective function by distorting the value of the transmitted $\bar{\mathbf{o}}^{L'}$. The precise definition of $\boldsymbol{\mathcal{H}}$ will be given in Section IV, while the considered fading distributions are discussed during the numerical evaluation in Section VII.

Assuming the wireless channel has sufficient capacity, optimizing both endpoints to perform source and channel encoding and decoding, under the standard paradigm of wireless communications, will indeed nullify the distortion on the received version of $\bar{\mathbf{o}}^{L'}$. The decoding output may then be passed to the $(L'+1)$-layer of the network as normal; note that the presence of the channel is effectively *hidden* from the neural network's perspective. This approach aligns more with the standard practices of both the wireless communications and machine learning paradigms, as each problem is treated individually and has indeed shown to exhibit satisfactory results [8], [14]. However, the above practice of optimizing the system for the reconstruction of the received signal may result in higher computational overheads under the following considerations:

1) Input reconstruction is not the objective of EI, rather the computation of an arbitrary function of it. Under this point of view, allocating computational resources to reconstructing intermediate variables is not always the most efficient way of solving the problem. In fact, one may regard GOC as a particular instance of lossy compression between the (unseen) target value $\mathbf{o}$ and its estimation $\hat{\mathbf{o}}$, where $\mathfrak{J}(\mathbf{o}, \hat{\mathbf{o}})$ plays the role of the distortion metric. From an information-theoretic perspective, variations of the distortion-rate functions may then be studied, as proposed by [23], indicating that the channel rate required to transmit intermediate variables, while ensuring a desired error threshold, is typically lower than the actual channel capacity which assumes reconstruction with arbitrarily small error probabilities.
2) From an engineering perspective, since the received signal is to be fed to subsequent neural network layers, perfect reconstruction may not be required. The employed neural network architectures are commonly designed to account for noisy inputs in light of the inherent stochasticity of inference problems. Besides, manually injecting noise to activations of intermediate layers, both during training [24], [25] and inference [26], [27], is known to enhance the model's regularization and uncertainty estimation properties.
3) Finally, the wireless channel, which can be regarded as a (naturally stochastic) function over the transmitted data, imposes its own computations. While this function is not, in general, controllable by the E2E system, OAC approaches leverage the superimposition of wireless signals to implement certain families of computational functions on top of the wireless medium[2]. Interestingly, the controllability offered by emerging MS technological solutions has the potential to enable more elaborate OAC, essentially offloading computations from the communication network endpoints.

III. Relevant State-of-the-Art

The implementation of DNNs at the transceivers has been first studied under the Joint Source Channel Coding (JSCC) paradigm, with results that outperform traditional communication systems due to the neural networks' ability to learn useful patterns from the data and channel distributions [29], [30]. JSCC approaches develop encoders (and respective decoders) that encode input data while simultaneously accounting for channel knowledge; this is different from the disjoint source and channel coding schemes pioneered by Shannon's separation theorem [9]. However, JSCC is limited to data reconstruction as an objective, where the RX aims to obtain an identical copy of the input. Conversely, deep semantic communication approaches [31] endow the communication system with the purpose of transmitting the meaning of the data, rather than its bit representations. The encoders and decoders, therefore, aim to extract informative representations, where the information is quantified by arbitrary cost functions which can be interpreted as different GOC objectives. Notably, the DeepSC architecture of [8] proposes a DNN approach with separate source and channel coding sub-modules that are trained independently. The channel encoder and decoder are trained to maximize an MI objective, which, on the one hand, is difficult to evaluate analytically for fading channels, while, on the other hand, maximizing the MI additionally implies that the effects of the channel are to be negated instead of being used for computations. In [10], a GOC approach with separate source and channel coding modules was developed for image retrieval under AWGN and Rayleigh fading, which further exemplified the benefits of separate training of each component rather than E2E. However, this training is possible due to the inserted rate-like part of the loss function that also treats the channel as noise, instead of a computational entity. Besides, the idea of DNN splitting for EI tasks has been investigated in [6], [20] from the information bottleneck viewpoint, to derive optimal network partitioning in uncontrollable wireless channels.

Traditionally, OAC methods have been developed to compute aggregate functions over multiple-access channels based on the superpositions of signals [32], and have been utilized

[2] In this paper, we make the distinction between "AirComp," as presented in [28], and OAC, despite borrowing similar naming for the transmission paradigms. AirComp specifically addresses nomographic functions and requires predefined analytical functions at the transceivers to compute them, finding particular applicability in multiple access and federated learning systems. Instead, we use the term OAC to refer to any effects of the propagation environment onto traveling signals, as long as the effects are controllable and are made toward a computational objective.

for federated learning tasks, as in [33]. Both of these OAC approaches, however, focused on computing a limited, yet useful family of analytical functions that are fundamentally different from the computations that take place in hidden neural network layers. More relevant to the present work is the methodology of [7], where the wireless channel is treated as a hidden network layer encompassed by DNN layers at the transceivers. While this E2E treatment can be utilized under the context of GOC, the fact that the computations imposed by wireless channels are not controllable in the absence of any MS flavor, provides limited benefits. In essence, the proposed MINN framework in this paper is general enough to accommodate this setup as a special case, once the MS-induced links are ignored in the overall system model. In the numerical evaluation section later on, we compare with this variation to illustrate the benefits of integrating MSs as hidden layers for effective mixed digital/analog computation, exemplifying the benefits of OAC.

The joint consideration of MSs and deep learning is a growing body of research. Apart from works that introduce deep learning algorithms to control RISs [34]–[36] or SIM [37], cascaded MSs have been introduced as DNN layers of diffractive implementations in [15]–[17], [38], toward analog computing hardware that is envisioned to exhibit notable benefits in computational speed and power consumption. In fact, developed SIM prototypes, such as those in [15]–[17], [38] as well as the references in [39], [40] have demonstrated, under controlled, quasi-free-space environments, that MSs with fixed configurations can be constructed with fully passive metamaterial [15], [38], which greatly reduces manufacturing and operational costs, whereas reconfigurable prototypes in high frequency bands [16], [41] may offer compact structures even when accounting for integrated controller hardware. In this paper, we are motivated by all-optical neural network implementations, but our framework is differentiated by the fact that the SIM layers are assumed to reside within the wireless environment, so that the E2E system needs to account for time-varying wireless fading when passing information between the SIM and digital network layers at the TX/RX endpoints [42]. It is noted that purely optical DNNs have the limitation that the input data needs to be transferred to the RF domain via techniques like holography, illumination, or traditional modulation, which are currently impractical for real-life deployment of such architectures. Finally, in the context of wireless networks, other works have capitalized on sophisticated designs of the responses of the elements of MSs to implement wave-domain signal processing [43], [44] or Multi-access Edge Computing (MEC) [45] tasks.

Regarding approaches that specifically consider GOC or semantic communication problems with the inclusion of MSs, a GOC approach was recently introduced in [13], according to which, the DNN layers at the transceivers were implemented via SIM layers, in contrast to having the SIM as part of the environment, as proposed by this work. Indeed, performing DNN computations at the RF regime even at the endpoints has the aforementioned benefits, however, the hardware design of such transceivers is far from trivial to be implemented by low-cost IoT devices. Besides, an MS placed directly inside the wireless environment may offer more precise control of the propagation medium. Additionally, SIMs-based DNNs, such as those considered in [13], [15], [38], assume the input data to be readily found in the wave domain via RF-responsive material, so that single-antenna TXs illuminate them to generate the input signals. This practice is, however, restrictive in normal EI scenarios, where the data is already digital, therefore it calls for a joint investigation of Multiple-Input Multiple-Output (MIMO) transmission and OAC strategies, as considered herein by the MINN paradigm.

Finally, semantic communications were performed through the assistance of an RIS placed at the environment in [14]. In that approach, the RIS was optimized to maximize an equivalent Signal-to-Noise Ratio (SNR) objective, similar to [8], whereas we herein propose to treat the problem in an E2E manner. In the numerical evaluation of Section VII, we include a baseline where the RIS, alongside other components, are optimized with respect to the achievable Shannon rate, and we show that this approach is less effective compared to our E2E treatment under the considered system, especially in the low-SNR regime. Despite the related literature of E2E architectures for EI that potentially take advantage of MS capabilities, to the best of our knowledge, the proposed MINN framework is the first work to highlight the importance of treating the MS-enabled smart wireless channel as a favorable computational machinery embedded inside an E2E DNN architecture.

IV. System and MS Components Modeling

A. System and Received Signal Models

We consider the uplink of a point-to-point MIMO communication system, where a TX equipped with N_t transmit antennas wishes to transmit its data to an N_r-antenna RX on a frame-by-frame basis, where the frames are indexed as $t = 1, 2, \ldots$ and are possibly unevenly spaced. This communication is enhanced via an MS (either an RIS or SIM), deployed as a standalone node in the wireless environment, whose configuration may change at every discrete time step t upon the command of an abstract controller unit. Without loss of generality, let us assume that the SIM is consisted of M thin diffractive layers, each with N_m unit elements, so that it contains $N_{\text{SIM}} \triangleq MN_m$ phase-tunable elements in total. For ease of notation and to present a comprehensive system model, we will additionally use N_m to denote the number of RIS tunable elements. Furthermore, let us define the Channel Frequency Response (CFR) matrices at each t-th time instance for the TX-RX, the TX-MS, and the MS-RX links as $\mathbf{H}_{\text{D}}(t) \in \mathbb{C}^{N_r \times N_t}$, $\mathbf{H}_1(t) \in \mathbb{C}^{N_t \times N_m}$, and $\mathbf{H}_2(t) \in \mathbb{C}^{N_r \times N_m}$, respectively. The transmitted signal is expressed as $\mathbf{s}(t) \in \mathbb{C}^{N_t \times 1}$, which satisfies a power budget constraint $P \triangleq \mathbb{E}[\|\mathbf{s}(t)\|]$. In fact, we suggest that $\mathbf{s}(t)$ represents both the intended, source-coded, and modulated data stream (according to the MIMO spatial multiplexing principle, the number of data symbols need to be $d \leq \min\{N_t, N_r\}$), as well as potential beamforming weights, without making any specific assumptions about the underlying procedures that produced the transmitted signal or the distribution of symbols.

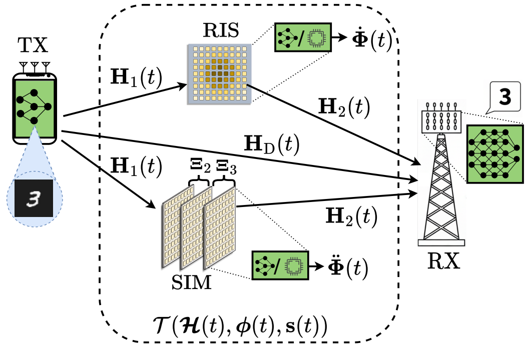
TX
RIS
$\dot{\mathbf{\Phi}}(t)$
$\mathbf{H}_1(t)$
$\mathbf{H}_2(t)$
3
$\mathbf{H}_\mathrm{D}(t)$
$\mathbf{H}_1(t)$
$\Xi_2\,\Xi_3$
$\mathbf{H}_2(t)$
3
$\ddot{\mathbf{\Phi}}(t)$
RX
SIM
$\mathcal{T}(\boldsymbol{\mathcal{H}}(t), \boldsymbol{\phi}(t), \mathbf{s}(t))$

Fig. 2. The considered MIMO system model used for EI incorporating either an RIS or a SIM device. Either of the MS structures includes a DNN-based controller or a basic processing unit to store or update its fixed configuration.

During each t-th frame transmission, the MS is characterized by its controllable *phase configuration* vector $\boldsymbol{\omega}(t) \in \mathbb{C}^{N_m \times 1}$ in the case of an RIS and $\boldsymbol{\omega}(t) \in \mathbb{C}^{N_\mathrm{SIM} \times 1}$ in the case of SIM, while its resulting *response configuration* is modeled for the idealized case of unit amplitude as $\boldsymbol{\phi}(t) \triangleq \exp(-\jmath\boldsymbol{\omega}(t))$. The effects of the responses of the metamaterials in the cascaded channel are captured via the matrix $\mathbf{\Phi}(t) \in \mathbb{C}^{N_m \times N_m}$, the structure of which, will be detailed in the next subsection. In the remainder of this work, $\mathbf{\Phi}(t)$, $\boldsymbol{\phi}(t)$, and $\boldsymbol{\omega}(t)$ are used as generic notation to describe any of the RIS and SIM cases, while device-specific notation is introduced wherever needed. Using the above, the baseband received signal at the RX antennas is expressed as follows:

$$\mathbf{y}(t) \triangleq \left(\mathbf{H}_\mathrm{D}(t) + \mathbf{H}_2(t)\mathbf{\Phi}(t)\mathbf{H}_1^\dagger(t)\right)\mathbf{s}(t) + \tilde{\mathbf{n}} \tag{4}$$

$$\triangleq \mathcal{T}(\boldsymbol{\mathcal{H}}(t), \boldsymbol{\phi}(t), \mathbf{s}(t)), \tag{5}$$

where $\tilde{\mathbf{n}} \in \mathbb{C}^{N_r \times 1}$ denotes the AWGN at the RX, comprising of independent and identically distributed (i.i.d.) samples drawn from the standard complex normal distribution $\mathcal{CN}(0, \sigma^2)$. In the sequel, we will be making use of the transmission function $\mathcal{T}(\boldsymbol{\mathcal{H}}(t), \boldsymbol{\phi}(t), \mathbf{s}(t))$ as an abstraction, emphasizing that the wireless medium is treated as a programmable computation. In this definition, we use the notation $\boldsymbol{\mathcal{H}}(t) \triangleq \{\mathbf{H}_\mathrm{D}(t), \mathbf{H}_1(t), \mathbf{H}_2(t)\}$ for the instantaneous Channel State Information (CSI), which is assumed to be readily available to all system nodes. Obviously, this availability implies a recurring channel estimation phase at each t-th time step, which may be a challenging prerequisite (see [35] and references therein). Nevertheless, this assumption allows the focus of this work to be on the training and evaluation of the proposed MINN architecture. Logical future extensions could incorporate the channel estimation phase in the DNN transceiver modules themselves, following ISAC principles [11]. Alternatively, channel-agnostic variations of transceivers will be also proposed and evaluated in the following sections to illustrate the performance trade-offs when integrating MSs as over-the-air neural network layers. The overall system model is depicted in Fig. 2, where either the RIS- or SIM-enabled links $\mathbf{H}_1(t)$ and $\mathbf{H}_2(t)$ are illustrated for each scenario.

B. RIS and SIM Models

Considering first an RIS, let its phase configuration vector at time t be denoted as $\dot{\boldsymbol{\omega}}(t) \triangleq [\dot{\omega}_1(t), \dot{\omega}_2(t), \ldots, \dot{\omega}_{N_m}(t)]^\top$ ($\equiv \boldsymbol{\omega}(t)$ in (5)), so that the phase state of its n-th unit element ($n = 1, 2, \ldots, N_m$) is expressed as $\dot{\omega}_n(t) \in [0, 2\pi)$. Then, the induced weights at the response configuration vector are given as $\dot{\boldsymbol{\phi}}(t) \triangleq \exp(-\jmath\dot{\boldsymbol{\omega}}(t))$ ($\equiv \boldsymbol{\phi}(t)$). In this case, using the diagonal matrix definition $\dot{\mathbf{\Phi}}(t) \triangleq \mathrm{diag}(\dot{\boldsymbol{\phi}}(t)) \in \mathbb{C}^{N_m \times N_m}$, it holds that $\mathbf{\Phi}(t) \equiv \dot{\mathbf{\Phi}}(t)$ in (4).

Proceeding to the introduction of the SIM into the system model, we first assume that all M layers ($m = 1, 2, \ldots, M$) are closely stacked and aligned parallel to each other, with their shared normal vector oriented perpendicular to the line connecting the TX and RX positions. Under this placement, the signal from the TX arrives at the first layer of the SIM, undergoes diffraction and controllable phase shifting by the consecutive $M - 1$ layers, before being finally diffracted toward the RX. Let us define the distance between any consecutive MS layers as d_M and the area of each unit element as S_M. Due to the compact placement of the layers, the layer-to-layer propagation can be accurately modeled via the Rayleigh-Sommerfeld diffraction equation [12], [13]. Namely, we define the propagation coefficient matrix from each $(m-1)$-th to the m-th layer as $\Xi_m \in \mathbb{C}^{N_m \times N_m}$, so that its (n, n')-th entry $(n, n' = 1, 2, \ldots, N_m)$ includes the propagation gain between the (arbitrarily ordered) n-th unit element of the $(m - 1)$-th layer and the n'-th element of the next layer, as follows [12]:

$$[\Xi_m]_{n,n'} \triangleq \frac{d_M S_M}{(d_{n,n'})^2}\left(\frac{1}{2\pi d_{n,n'}} - \frac{\jmath}{\lambda}\right)\exp(\jmath 2\pi d_{n,n'}), \tag{6}$$

where $d_{n,n'}$ denotes the distance between the centers of the n-th and n'-th elements, and λ is the carrier frequency.

Apart from diffracting, each n-th element of each m-th SIM layer introduces a controllable weight, similar to the RIS modeling, denoted as $[\ddot{\boldsymbol{\phi}}_m(t)]_n \triangleq \exp(-\jmath\ddot{\omega}_n^m(t))$, with $\ddot{\omega}_n^m(t) \in [0, 2\pi)$ being the phase state of the n-th unit element of the m-th layer. We also introduce $\ddot{\boldsymbol{\phi}}_m(t)$ including the *response configuration* at the m-th layer, the overall response configuration $\ddot{\boldsymbol{\phi}}(t) \triangleq [\ddot{\boldsymbol{\phi}}_1^\top(t), \ddot{\boldsymbol{\phi}}_2^\top(t), \ldots, \ddot{\boldsymbol{\phi}}_M^\top(t)]^\top \in \mathbb{C}^{N_\mathrm{SIM} \times 1}(\equiv \boldsymbol{\phi}(t))$, and the *phase configuration* vector of the SIM $\ddot{\boldsymbol{\omega}}(t) \triangleq [\ddot{\omega}_1^1(t), \ldots, \ddot{\omega}_{N_m}^1(t), \ldots, \ddot{\omega}_1^M(t), \ldots, \ddot{\omega}_{N_m}^M(t)]^\top \in \mathbb{C}^{N_\mathrm{SIM} \times 1}$. By defining $\ddot{\mathbf{\Phi}}_m(t) \triangleq \mathrm{diag}(\ddot{\boldsymbol{\phi}}_m(t))$, the overall SIM response can be mathematically expressed via the following matrix [42]:

$$\ddot{\mathbf{\Phi}}(t) \triangleq \left(\prod_{m=M}^{2} \ddot{\mathbf{\Phi}}_m(t)\Xi_m\right)\ddot{\mathbf{\Phi}}_1(t) \in \mathbb{C}^{N_m \times N_m}. \tag{7}$$

Note that, for $\mathbf{\Phi}(t) \equiv \ddot{\mathbf{\Phi}}(t)$, (4) holds for the SIM case.

Revisiting the previously introduced generic notations of $\mathbf{\Phi}(t)$ and $\boldsymbol{\omega}(t)$, they can now be expressed concretely for each of the RIS and SIM cases as $\mathbf{\Phi}(t) \in \{\dot{\mathbf{\Phi}}(t), \ddot{\mathbf{\Phi}}(t)\}$ and $\boldsymbol{\omega}(t) \in \{\dot{\boldsymbol{\omega}}(t), \ddot{\boldsymbol{\omega}}(t)\}$. In the rest of this paper, $\mathbf{\Phi}(t)$, $\boldsymbol{\omega}(t)$, and $\boldsymbol{\phi}(t)$ will be used when the underlying operations are agnostic of the MS type, while $\dot{\mathbf{\Phi}}(t)$, $\ddot{\mathbf{\Phi}}(t)$, and their respective vectors will be explicitly utilized when the operations need to discriminate between RISs and SIM.

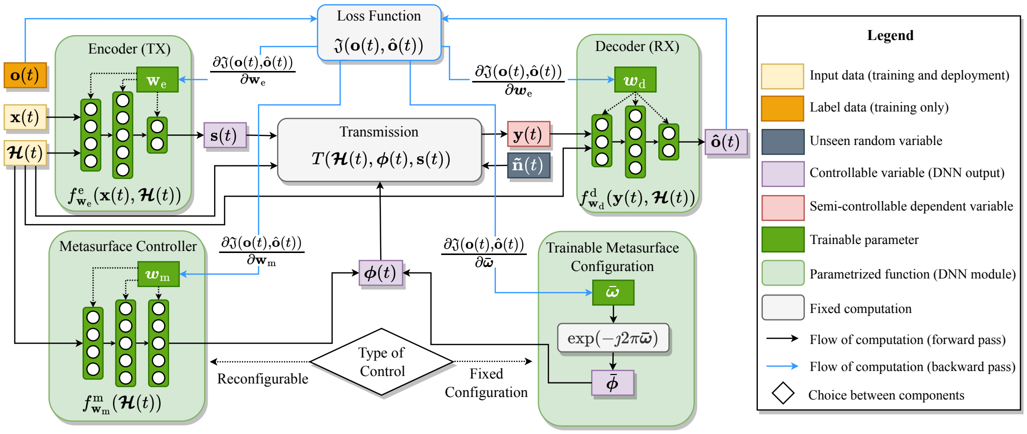
Loss Function
$\mathfrak{J}(\mathbf{o}(t), \hat{\mathbf{o}}(t))$
Encoder (TX)
$\mathbf{o}(t)$
$\mathbf{x}(t)$
$\boldsymbol{\mathcal{H}}(t)$
$\mathbf{w}_\mathrm{e}$
$\frac{\partial\mathfrak{J}(\mathbf{o}(t),\hat{\mathbf{o}}(t))}{\partial\mathbf{w}_\mathrm{e}}$
$f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))$
$\mathbf{s}(t)$
Transmission
$T(\boldsymbol{\mathcal{H}}(t), \boldsymbol{\phi}(t), \mathbf{s}(t))$
$\mathbf{y}(t)$
$\tilde{\mathbf{n}}(t)$
Decoder (RX)
$\frac{\partial\mathfrak{J}(\mathbf{o}(t),\hat{\mathbf{o}}(t))}{\partial\boldsymbol{w}_\mathrm{e}}$
$\boldsymbol{w}_\mathrm{d}$
$f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\mathbf{y}(t), \boldsymbol{\mathcal{H}}(t))$
$\hat{\mathbf{o}}(t)$
Metasurface Controller
$\frac{\partial\mathfrak{J}(\mathbf{o}(t),\hat{\mathbf{o}}(t))}{\partial\mathbf{w}_\mathrm{m}}$
$\boldsymbol{w}_\mathrm{m}$
$f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}(\boldsymbol{\mathcal{H}}(t))$
$\boldsymbol{\phi}(t)$
Type of Control
Reconfigurable
Fixed Configuration
$\frac{\partial\mathfrak{J}(\mathbf{o}(t),\hat{\mathbf{o}}(t))}{\partial\bar{\boldsymbol{\omega}}}$
Trainable Metasurface Configuration
$\bar{\boldsymbol{\omega}}$
$\exp(-\jmath 2\pi\bar{\boldsymbol{\omega}})$
$\bar{\boldsymbol{\phi}}$
Legend
Input data (training and deployment)
Label data (training only)
Unseen random variable
Controllable variable (DNN output)
Semi-controllable dependent variable
Trainable parameter
Parametrized function (DNN module)
Fixed computation
Flow of computation (forward pass)
Flow of computation (backward pass)
Choice between components

Fig. 3. Block diagram and computation flow for the proposed E2E GOC framework where the metasurface-parametrizable channel acts as an intermediate DNN component. Both the cases of reconfigurable and static metasurfaces are included, entailing different procedures during the forward and backward passes.

V. MSs-Integrated Neural Networks (MINNs)

A. Transceiver Modules

As initially discussed in Section II-C, EI entails two computational modules collocated at the transceiver endpoints that are implemented with digital processing hardware. To implement the *"infer-while-transmitting"* methodology, the TX utilizes an encoder DNN, $f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\cdot)$, providing the output $\mathbf{s}(t)$, while the RX operates a decoder DNN, $f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\cdot)$, deriving the output $\hat{\mathbf{o}}(t)$. Those blocks are tasked with performing compression, encoding and decoding, error resilience and correction, and potential transmit and receive beamforming alongside probabilistic inference. The exact layer architecture of those models is purposely left unspecified at this stage as a practitioner's choice, depending, in general, on: *i*) the nature of the wireless environment; *ii*) the type of input and target data; *iii*) computational capabilities of the transceivers' hardware; as well as *iv*) the current state-of-the-art. We only note that different sub-modules may be used for each of the above operations, while, typically for uplink scenarios, $f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\cdot)$ can be implemented with larger DNN structures due to the constant power supply at base stations. In addition, regardless of the choice of neural network, we impose a final fixed post-processing step at the encoder's output $\mathbf{s}(t)$ to satisfy the TX system's power budget, as follows:

$$\mathbf{s}(t) \leftarrow \sqrt{P}\frac{\mathbf{s}(t)}{\|\mathbf{s}(t)\|}. \tag{8}$$

Considering the concrete input arguments of the encoder functions, two different variations may be defined depending on whether CSI is available to each of the endpoints.

1) Channel-Agnostic Transceivers: An instance of the data variables $\mathbf{x}(t)$ is observed at the TX, that is passed to the encoder to construct the transmitted signal, while the decoder DNN observes the received signal and performs an estimate of the unseen target variable $\mathbf{o}(t)$; this can be described as:

$$\mathbf{s}(t) = f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\mathbf{x}(t)), \tag{9}$$

$$\hat{\mathbf{o}}(t) = f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\mathbf{y}(t)). \tag{10}$$

Since no CSI is used by the endpoints, we highlight the similarity of this design to source-only coding, despite the fact that the encoder may need to add redundancy in the transmitted signal, which is traditionally considered as channel coding. Evidently, both processes need to guarantee that the inference procedure performs sufficiently, irrespective of the current channel conditions, which may be a demanding request. Nevertheless, not requiring CSI is a strong simplification of the system architecture, therefore, it is included later on in our investigations in this paper.

2) Channel-Aware Transceivers: Let us assume a quasi-static fading channel and a channel estimation procedure that takes place within each t-th channel frame before data transmission, based on which the TX and RX modules obtain accurate estimates of the CFR matrices $\boldsymbol{\mathcal{H}}(t)$. Each module may receive $\boldsymbol{\mathcal{H}}(t)$ as an additional input, yielding respectively the following representations for the encoder/decoder DNNs:

$$\mathbf{s}(t) = f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t)), \tag{11}$$

$$\hat{\mathbf{o}}(t) = f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\mathbf{y}(t), \boldsymbol{\mathcal{H}}(t)). \tag{12}$$

Endowing the TX/RX modules with CSI leads to more resilient transmission schemes that closely resemble JSCC [46]. The main difference lies in that JSCC focuses on estimating $\mathbf{x}(t)$, while EI deals with approximating $\mathbf{o}(t) = l(\mathbf{x}(t))$[3]. In this paper, we assume that channel estimation takes place transparently before every transmission, both during training and inference, and results in noise-free estimations of $\boldsymbol{\mathcal{H}}(t)$. Accounting for noisy estimates or even incorporate the estimation in the procedure under ISAC paradigms lead exciting research directions, which we will study in future works.

B. Control Module for Reconfigurable Metasurfaces

When CSI is available, as in most wireless communication settings, the MS changes its response configuration at every transmission frame to optimize the system's objective [47]. To incorporate this mode of operation into our E2E architecture, $\boldsymbol{\phi}(t)$ is treated as a controllable variable that is the output

[3] Those two problems can be considered equivalent by setting the mapping function $l(\cdot)$ to be the identity function, yielding $\mathbf{o}(t) = \mathbf{x}(t)$, and adopting MSE as the objective function $\mathfrak{J}(\cdot,\cdot)$. As a result, EI is a more general problem formulation than communications which focus on data reconstruction.

of a third digital DNN module. Specifically, we define the *metasurface controller* as the following neural network:

$$\boldsymbol{\phi}(t) = f^{\mathrm{m}}_{\mathbf{w}_{\mathrm{m}}}(\boldsymbol{\mathcal{H}}(t)), \tag{13}$$

imposing that the final layer performs the operation $\boldsymbol{\phi}(t) = \exp(-\jmath\hat{\boldsymbol{\omega}})$, where $\hat{\boldsymbol{\omega}}$ denotes the outputs of the penultimate layer with elements lying in $[0, 2\pi)$. As stated before, either $\dot{\boldsymbol{\phi}}(t)$ or $\ddot{\boldsymbol{\phi}}(t)$ may be the actual output of the module depending of the selected type of MS, however, we keep the abstract notation of $\boldsymbol{\phi}(t)$ to provide a general framework. Under this viewpoint, the MS is a controllable entity that can be adapted dynamically to offer favorable wave-domain computation at every channel realization. This treatment allows for fine-grained control over the reprogrammability of the environment, at the cost of an additional neural network module and the associated hardware requirements. Plugging the three trained modules from expressions (9)–(13) onto the received signal in (4), we can derive the E2E inference model $\hat{\mathbf{o}}(t) = f^{\mathrm{r}}_{\mathbf{w}_{\mathrm{r}}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))$, for the two channel knowledge cases, as depicted in Fig.3, as follows:

$$\hat{\mathbf{o}}(t) = \underbrace{f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}\left(\mathcal{T}\left(\boldsymbol{\mathcal{H}}(t), f^{\mathrm{m}}_{\mathbf{w}_{\mathrm{m}}}(\boldsymbol{\mathcal{H}}(t)), f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t))\right)\right)}_{\text{channel-agnostic transceivers}}, \text{ or} \tag{14a}$$

$$\hat{\mathbf{o}}(t) = \underbrace{f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}\left(\mathcal{T}\left(\boldsymbol{\mathcal{H}}(t), f^{\mathrm{m}}_{\mathbf{w}_{\mathrm{m}}}(\boldsymbol{\mathcal{H}}(t)), f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))\right), \boldsymbol{\mathcal{H}}(t)\right)}_{\text{channel-aware transceivers}}, \tag{14b}$$

where the overall trainable weights of this reconfigurable architecture have been represented as $\mathbf{w}_{\mathrm{r}} \triangleq \{\mathbf{w}_{\mathrm{d}}, \mathbf{w}_{\mathrm{e}}, \mathbf{w}_{\mathrm{m}}\}$, which can be trained together under the same objective functions and backward passes, as it will be detailed in Section VI. Note that, in (14a), we have purposefully allowed the control module to be channel-aware. Such a system implies that the MS devices have sensing capabilities (e.g., [48]) and can thus acquire channel knowledge. The role of this system is discussed further in Section VI-B.

C. Metasurfaces with Trainable Fixed Response

As an alternative approach, one may choose to directly learn a fixed configuration for the MS; let us denote this as $\bar{\boldsymbol{\omega}}$. While the training process may iteratively evaluate multiple candidate values for $\bar{\boldsymbol{\omega}}$, once the training is complete, the learned configuration is equipped onto the MS to maintain a constant (static) response configuration $\boldsymbol{\phi}(t) \equiv \bar{\boldsymbol{\phi}} \triangleq \exp(-\jmath\bar{\boldsymbol{\omega}})$ over time, irrespective of the channel conditions or input data. This description is more akin to the idea that the effective phase configurations are treated similarly to DNN weights, as they too remain fixed after the completion of the training procedure, and are used to perform the same computational operations over varying input instances. The training procedure optimizes $\bar{\boldsymbol{\omega}}$ directly, i.e., its weights $\mathbf{w}_{\mathrm{s}} \triangleq \{\mathbf{w}_{\mathrm{d}}, \mathbf{w}_{\mathrm{e}}, \bar{\boldsymbol{\omega}}\}$, therefore, the E2E static architecture can be now expressed as follows:

$$\hat{\mathbf{o}}(t) = \underbrace{f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}\left(\mathcal{T}\left(\boldsymbol{\mathcal{H}}(t), \bar{\boldsymbol{\phi}}, f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t))\right)\right)}_{\text{channel-agnostic transceivers}}, \text{ or} \tag{15a}$$

$$\hat{\mathbf{o}}(t) = \underbrace{f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}\left(\mathcal{T}\left(\boldsymbol{\mathcal{H}}(t), \bar{\boldsymbol{\phi}}, f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))\right), \boldsymbol{\mathcal{H}}(t)\right)}_{\text{channel-aware transceivers}}. \tag{15b}$$

It is noted that, while reconfigurable MSs may offer more precise control in shaping the exact form of $\mathcal{T}(\cdot)$, the extra hidden layers required by the inclusion of the MS controller module may well hinder the training capabilities of the proposed MINN compared to the current variation. Besides, assuming wireless systems of reasonably limited variability, such as Line-of-Sight (LoS) dominant environments with fixed transceivers, static MS configurations may offer satisfactory performance. The next section addresses the systemic requirements for all variations of this section, while performance trade-offs are investigated under our numerical evaluations.

D. Analysis and Extensions

1) Multiple Transmissions per Inference Step: The inference scheme described above considers a single data transmission per inference step t, i.e., once $\mathbf{x}(t)$ is observed, it is encoded in its entirety onto $\mathbf{s}(t)$, which is transmitted within a single Transmission Time Interval (TTI) and then received as $\mathbf{y}(t)$, before the estimation $\hat{\mathbf{o}}(t)$ is obtained. Since the dimensions of $\mathbf{s}(t)$ and $\mathbf{y}(t)$ are constrained by the transceivers' antenna elements, in more challenging data sets, it might be desirable for the encoder to output larger vectors and transmit them over consecutive TTIs. Such an extension is easily accommodated by the MINN framework: assume that the channel fading remains quasi-static for a certain duration to allow τ transmissions under the same $\mathcal{H}(t)$ at a time step t. Using channel-aware transceivers as an example, the encoder produces a series of outputs $\mathbf{s}(t) = \{\mathbf{s}(t,1), \ldots, \mathbf{s}(t,\tau)\} = f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t))$, while the decoder observes all corresponding received signals $\mathbf{y}(t) = \{\mathbf{y}(t,1), \ldots, \mathbf{y}(t,\tau)\}$ before performing its forward pass to output $\hat{\mathbf{o}}(t) = f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}(\mathbf{y}(t))$. The MS control module, if present, decides on a single configuration $\boldsymbol{\phi}(t)$ for all τ transmissions, since the channel, being the input to $f^{\mathrm{m}}_{\mathbf{w}_{\mathrm{m}}}(\boldsymbol{\mathcal{H}}(t))$, remains fixed. In the reminder of the work, we will be considering single-TTI scenarios due to ease of notation, since derivations of multi-TTI cases are a matter of decomposing the involved vectors into their τ composite components. Besides, it is reasonable to assume that only a limited amount of TTIs might take place within a channel coherence block. For the most demanding cases, one might employ multiple coherence blocks so that the estimation for $\mathbf{x}(t)$ is obtained at a later frame $t' > t$. While a possible description of this process resembles the formulation discussed above, a thorough analysis would include considering time-evolving channel models and could possibly require purposely designed recurrent DNN architectures, therefore, its complete investigation is omitted for the shake of brevity.

2) Properties of MSs as Hidden Layers: From the received signal expression in (4), it is apparent that $\mathcal{T}(\cdot)$ is a linear transformation on $\mathbf{s}(t)$, irrespective of whether multiple SIM layers are used or a single RIS. Therefore, $\mathcal{T}(\cdot)$ acts a single hidden layer, without an activation function. Regardless, for the case of SIM, the effects of $\boldsymbol{\phi}(t)$ on $\mathbf{s}(t)$ are highly nonlinear due to the repeated multiplications of the overall SIM (7), which is itself multiplied with the non-diagonal matrix $\mathbf{H}_2(t)$. Therefore, increasing the number of layers and elements allows for more precise manipulation of the cascaded channel $\mathbf{H}_\mathrm{D}(t)+\mathbf{H}_2(t)\boldsymbol{\Phi}(t)\mathbf{H}_1^\dagger(t)$ of (4). The MINN framework, as presented in this paper, makes use of the established SIM propagation model followed by the community (e.g., [12], [13], [15]) that relies on fixed unit amplitude elements without any phase-amplitude relations.

Nevertheless, MSs with more intricate functionalities may be utilized, such as ones containing power amplification [49], [50]. As shown in [50], in such cases, the E2E signal model ceases to be a linear function over the input signal. Therefore, stacking multiple such RISs would result in MINNs where $\mathcal{T}(\cdot)$ is equivalent to deep network structures. Very recently, the work of [51] considered nonlinear amplifiers as part of MS elements to implement sigmoid-like activation functions in the RF domain, toward proving universal approximation [19] on extremely large MIMO systems seen as neural networks. However, this work considered only single-hidden-layer networks with static fading, making it less applicable to the comprehensive MINN framework. Since the practical maturity of such advanced MS designs remains yet limited and the E2E DNN would rely heavily on the hardware design used, we leave the investigation of such scenarios for future work.

We note instead that, E2E, the MINN framework does exhibit universal approximation properties since it includes the encoder and decoder modules, that are themselves universal approximators, as long as: *(i)* the SNR is reasonably high; and *(ii)* the channel fading does not destroy the information encoded in the transmitted signal. To ensure the last part, the MS effects on the cascaded channel must preserve or enhance the channel rank (or equivalent spectral properties). This observation provides a direction for MS optimization that is independent of the other DNN modules trained via SGD. Nevertheless, as suggested by the numerical evaluation that follows, where capacity-optimizing RIS, precoder, and combiners are used, simply enhancing the spectral properties of the cascaded channel is not sufficient for EI. Full-rank channels only ensure that information is not lost during transmission, whereas the E2E training on the EI objective $\mathcal{OP}_\mathrm{EI}$ is designed to perform computation that extracts features from data that are helpful for the intended inference task, therefore, offloading part of the computations onto the channel.

3) Digital and Over-the-Air Computing Nodes: The implementation of MINNs relies on traditional digital processors, over-the-air propagation effects, as well as analog RF circuitry. In detail, $f^{\mathrm{e}}_{\mathbf{w}_\mathrm{e}}(\cdot)$, $f^{\mathrm{d}}_{\mathbf{w}_\mathrm{d}}(\cdot)$, and $f^{\mathrm{m}}_{\mathbf{w}_\mathrm{m}}(\cdot)$, if present, are implemented through digital processors, while $\mathcal{T}(\cdot)$ is realized over the air. Interestingly, when MSs with static responses are considered within the MINN framework, the learned phase configurations require basic analog RF circuitry or even completely passive material (e.g., as in [15], [38]) for their implementation.

VI. MINN Training and Deployment

To perform neural network training at any of the wireless communication system nodes, a separate data collection step is carried out to generate a set of $|\mathcal{D}|$ labeled data instances: $\mathcal{D}\triangleq\{(\mathbf{x}_i,\mathbf{o}_i)\}_{i=1}^{|\mathcal{D}|}$. Let us also assume the availability of a set of $|\mathcal{C}|$ channel sample estimates (in respective coherent time instances): $\mathcal{C}\triangleq\{\boldsymbol{\mathcal{H}}(t)\}_{t=1}^{|\mathcal{C}|}$, not necessarily equally spaced. In this paper, we make the assumption that the channel realizations are conditionally independent[4] from $\mathcal{D}$'s data instances. It is noted that, while this is a rather lenient assumption, it is crucial in permitting the evaluation of the expectation in $\mathcal{OP}_\mathrm{EI}$'s objective via i.i.d. Monte Carlo samples.

A. Backpropagation Over the Wireless Channel

The training procedure can be described as a variation of the standard gradient descent approach for neural network training, with the inclusion of channel samples. To provide a comprehensive framework, let us use the generic parameter vector $\mathbf{w}_\mathrm{k}$ with $\mathrm{k}\in\{\mathrm{r},\mathrm{s}\}$, taking the form of either $\mathbf{w}_\mathrm{r}$ or $\mathbf{w}_\mathrm{s}$ depending on the choice of use of reconfigurable or static MSs. Similar to standard deep learning practices, our E2E MINN architecture may be optimized using SGD over the collected data and channel instances. Specifically, let us express the data-channel loss function as $\mathfrak{J}(\mathbf{o}_i,\hat{\mathbf{o}}_i)=\mathfrak{J}(\mathbf{o}_i,f^{\mathrm{k}}_{\mathbf{w}_\mathrm{k}}(\mathbf{x}(t),\boldsymbol{\mathcal{H}}(t)))$ to explicit show the dependence of the loss function on the instantaneous wireless channel conditions. To this end, leveraging the previous described conditional independence assumption, $\mathcal{OP}_\mathrm{EI}$'s objective can be approximated as follows:

$$\mathbb{E}_{\boldsymbol{\mathcal{H}}}[J(\mathbf{w}_\mathrm{k})]\cong\frac{1}{|\mathcal{C}||\mathcal{D}|}\sum_{t=1}^{|\mathcal{C}|}\sum_{i=1}^{|\mathcal{D}|}\mathfrak{J}(\mathbf{o}_i,f^{\mathrm{k}}_{\mathbf{w}_\mathrm{k}}(\mathbf{x}_i,\boldsymbol{\mathcal{H}}(t))). \tag{16}$$

In the online version of SGD, at every time t, one may select a single data point and channel instance to evaluate (16), and accordingly update the parameter vector as follows:

$$\mathbf{w}_\mathrm{k}\leftarrow\mathbf{w}_\mathrm{k}-\eta\nabla_{\mathbf{w}_\mathrm{k}}\mathfrak{J}(\mathbf{o}(t),f^{\mathrm{k}}_{\mathbf{w}_\mathrm{k}}(\mathbf{x}(t),\boldsymbol{\mathcal{H}}(t))), \tag{17}$$

for some chosen learning rate η, with the gradient at each case being defined by one of the two following expressions:

$$\nabla\mathfrak{J}=\underbrace{\left[\left[\frac{\partial\mathfrak{J}}{\partial\mathbf{w}_\mathrm{d}}\right]^\top,\left[\frac{\partial\mathfrak{J}}{\partial\mathbf{w}_\mathrm{e}}\right]^\top,\left[\frac{\partial\mathfrak{J}}{\partial\mathbf{w}_\mathrm{m}}\right]^\top\right]^\top}_{\text{reconfigurable metasurface}} \tag{18}$$

$$\nabla\mathfrak{J}=\underbrace{\left[\left[\frac{\partial\mathfrak{J}}{\partial\mathbf{w}_\mathrm{d}}\right]^\top,\left[\frac{\partial\mathfrak{J}}{\partial\mathbf{w}_\mathrm{e}}\right]^\top,\left[\frac{\partial\mathfrak{J}}{\partial\bar{\boldsymbol{\omega}}}\right]^\top\right]^\top}_{\text{metasurface with trainable fixed response}}. \tag{19}$$

[4] In certain scenarios, the data realizations and the statistical properties of the channel can be statistically dependent. For example, in a target detection system where the observations $\mathbf{x}_i$ contain sensory inputs, while $\mathbf{o}_i$ is a binary variable indicating the existence of a target in the area of interest, deep fading may be encountered more often when a target is subject to signal blockages. In such cases, the two collection processes of channel measurements and observed data must be synchronous, and a more detailed formulation of the EI objective is required. However, the inference problem itself may be potentially computationally easier, since the CSI observation provides additional information regarding the target value.

Algorithm 1 Training of the Proposed E2E MINN

```
1:  Construct DNN weight vector $\mathbf{w}$ as one of the following:
        i) $\mathbf{w}_\mathrm{k} = \mathrm{concat}(\mathbf{w}_\mathrm{d}, \mathbf{w}_\mathrm{e}, \mathbf{w}_\mathrm{m})$.        ▷ $\mathbf{w}_\mathrm{k} \leftarrow \mathbf{w}_\mathrm{r}$
        ii) $\mathbf{w}_\mathrm{k} = \mathrm{concat}(\mathbf{w}_\mathrm{d}, \mathbf{w}_\mathrm{e}, \bar{\boldsymbol{\omega}})$.        ▷ $\mathbf{w}_\mathrm{k} \leftarrow \mathbf{w}_\mathrm{s}$
2:  Initialize $\mathbf{w}$ randomly.
3:  for $t = 1, 2, \ldots,$ until convergence do
4:      Sample $(\mathbf{x}(t), \mathbf{o}(t))$ from $\mathcal{D}$.
5:      Sample $\boldsymbol{\mathcal{H}}(t)$ from $\mathcal{C}$.
6:      Compute $\mathbf{s}(t)$ using one of the following:
          i) $\mathbf{s}(t) = f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\mathbf{x}(t))$.        ▷ Eq. (9)
          ii) $\mathbf{s}(t) = f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))$.        ▷ Eq. (11)
7:      Compute $\boldsymbol{\phi}(t)$ using one of the following:
          i) $\boldsymbol{\phi}(t) = f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}(\boldsymbol{\mathcal{H}}(t))$.        ▷ Eq. (13)
          ii) $\boldsymbol{\phi}(t) = \bar{\boldsymbol{\phi}} = \exp(-\jmath\bar{\boldsymbol{\omega}})$.
8:      Transmit $\mathbf{s}(t)$ to receive $\mathbf{y}(t)$:
          $\mathbf{y}(t) = \mathcal{T}(\boldsymbol{\mathcal{H}}(t), \boldsymbol{\phi}(t), \mathbf{s}(t))$.        ▷ Eq. (5)
9:      Compute $\hat{\mathbf{o}}(t)$ using one of the following:
          i) $\hat{\mathbf{o}}(t) = f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\mathbf{y}(t))$.        ▷ Eq. (10)
          ii) $\hat{\mathbf{o}}(t) = f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}(\mathbf{y}(t), \boldsymbol{\mathcal{H}}(t))$.        ▷ Eq. (12)
10:     Set $\mathbf{w}_\mathrm{k} \leftarrow \mathbf{w}_\mathrm{k} - \eta \nabla_{\mathbf{w}_\mathrm{k}} \mathfrak{J}(\mathbf{o}(t), f^\mathrm{k}_{\mathbf{w}_\mathrm{k}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t)))$.
11: end for
12: return $\mathbf{w}$
```

Under the i.i.d. sampling assumption, the consecutive evaluations of the gradient of the objective function in (17) at each training instance t are unbiased estimators of the true gradient of the objective in (16). Therefore, following the stochastic approximation framework, the repetition of this procedure will converge to the true value of the expectation with probability 1 up to a precision of $O(\eta)$ around it, using constant step size [52]. The complete training procedure is detailed in Algorithm 1, which supports all variations of channel-agnostic/-aware transceivers, static/reconfigurable MS controllers, and RIS/SIM structure. Lines 6-9 implement our MINN architecture as defined in (14) and (15). Naturally, batched gradient descent versions may be used alongside more elaborate gradient updates, such as momentum, weight decay (regularization), and adaptive rates [53], however, such implementation details have been left out for ease of presentation.

The crux of the training procedure is the gradient update mechanism of (17). Since (14) and (15) are differentiable operations with respect to $\mathbf{w}_\mathrm{s}$ or $\mathbf{w}_\mathrm{r}$, the partial derivatives may be computed via automatic differentiation tools, by applying the chain rule on the underlying computational graph. Regardless, for the shake of completeness, we provide the derivations for the partial derivatives of the various modules, however, treating the implementation-defined derivatives of the classical neural network components (i.e., $\partial f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}/\partial \mathbf{w}_\mathrm{e}$, $\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}/\partial \mathbf{w}_\mathrm{d}$, $\partial f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}/\partial \mathbf{w}_\mathrm{m}$, and $\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}/\partial \mathbf{y}(t)$) as known. Continuing, we will make use of the identity $\mathrm{vec}(\mathbf{AXB}) = (\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X})$ and that, for an n-element vector $\mathbf{x}$ and $\mathbf{X} = \mathrm{diag}(\mathbf{x})$, the vectorization operation on $\mathbf{X}$ can be expressed using matrix operations as $\mathrm{vec}(\mathbf{X}) = \mathbf{D}\mathbf{x}$, where $\mathbf{D} \triangleq [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_n]$ is an $n^2 \times n$ matrix used for selecting the diagonal elements, in which $\mathbf{D}_i$ is an $n \times n$ matrix with binary elements having 1 at its (i,i)-th element and 0 elsewhere.

For the case of a reconfigurable MS (either an RIS or SIM), $\hat{\mathbf{o}}$ is computed via (14). By applying backwards propagation, the following derivations are deduced:

$$\frac{\partial \mathfrak{J}}{\partial \mathbf{w}_\mathrm{d}} = \frac{\partial \mathfrak{J}}{\partial \hat{\mathbf{o}}(t)} \cdot \frac{\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}}{\partial \mathbf{w}_\mathrm{d}}, \tag{20}$$

$$\frac{\partial \mathfrak{J}}{\partial \mathbf{w}_\mathrm{m}} = \frac{\partial \mathfrak{J}}{\partial \hat{\mathbf{o}}(t)} \cdot \frac{\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}}{\partial \mathbf{y}(t)} \cdot \frac{\partial \mathbf{y}(t)}{\partial f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}} \cdot \frac{\partial f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}}{\partial \mathbf{w}_\mathrm{m}}, \tag{21}$$

$$\frac{\partial \mathfrak{J}}{\partial \mathbf{w}_\mathrm{e}} = \frac{\partial \mathfrak{J}}{\partial \hat{\mathbf{o}}(t)} \cdot \frac{\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}}{\partial \mathbf{y}(t)} \cdot \frac{\partial \mathbf{y}(t)}{\partial f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}} \cdot \frac{\partial f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}}{\partial \mathbf{w}_\mathrm{e}}, \tag{22}$$

where $\partial \mathfrak{J}/\partial \hat{\mathbf{o}}(t)$ expresses the gradient of the problem-defined loss function with respect to the network's output, which, for the example of the CE loss of (2), is computed as $-\mathbf{o}(t)/\hat{\mathbf{o}}(t)$. The remaining terms are defined as follows:

$$\frac{\partial \mathbf{y}(t)}{\partial f^\mathrm{e}_{\mathbf{w}_\mathrm{e}}} = \mathbf{H}_2(t)\boldsymbol{\Phi}(t)\mathbf{H}_1^\dagger(t) + \mathbf{H}_\mathrm{D}(t), \tag{23}$$

$$\frac{\partial \mathbf{y}(t)}{\partial f^\mathrm{m}_{\mathbf{w}_\mathrm{m}}} = \frac{\partial \mathbf{y}(t)}{\partial \boldsymbol{\phi}(t)} = \big((\mathbf{s}^\top(t)\mathbf{H}_1^*(t)) \otimes \mathbf{H}_2(t)\big)\mathbf{D}, \tag{24}$$

with $\mathbf{D} \in [0,1]^{N_m^2 \times N_m}$ being a block-diagonal selection matrix.

In the fixed-configuration RIS case, $\hat{\mathbf{o}}$ is computed via (15). To highlight the particular case of SIM, we introduce the notations $\bar{\boldsymbol{\omega}} \equiv \bar{\boldsymbol{\omega}}_\mathrm{RIS} \in [0, 2\pi)^{N_m}$ and $\bar{\boldsymbol{\phi}} \equiv \bar{\boldsymbol{\phi}}_\mathrm{RIS} \triangleq \exp(-\jmath\bar{\boldsymbol{\omega}}_\mathrm{RIS})$. Since $\partial\mathfrak{J}/\partial\mathbf{w}_\mathrm{e}$ and $\partial\mathfrak{J}/\partial\mathbf{w}_\mathrm{e}$ remain the same, it holds:

$$\frac{\partial \mathfrak{J}}{\partial \bar{\boldsymbol{\omega}}} = \frac{\partial \mathfrak{J}}{\partial \bar{\boldsymbol{\omega}}_\mathrm{RIS}} = \frac{\partial \mathfrak{J}}{\partial \hat{\mathbf{o}}(t)} \cdot \frac{\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}}{\partial \mathbf{y}(t)} \cdot \frac{\partial \mathbf{y}(t)}{\partial \bar{\boldsymbol{\phi}}_\mathrm{RIS}} \cdot \frac{\partial \bar{\boldsymbol{\phi}}_\mathrm{RIS}}{\partial \bar{\boldsymbol{\omega}}_\mathrm{RIS}}, \tag{25}$$

where $\partial \mathbf{y}(t)/\partial \bar{\boldsymbol{\phi}}_\mathrm{RIS}$ can be computed as in (24), while $\partial \bar{\boldsymbol{\phi}}_\mathrm{RIS}/\partial \bar{\boldsymbol{\omega}}_\mathrm{RIS} = -\jmath \exp(-\jmath\bar{\boldsymbol{\omega}}_\mathrm{RIS})$.

For the fixed-configuration SIM case, we introduce the notations $\bar{\boldsymbol{\omega}} \equiv \bar{\boldsymbol{\omega}}_\mathrm{SIM} \in [0, 2\pi)^{N_\mathrm{SIM}}$ and $\bar{\boldsymbol{\phi}} \equiv \bar{\boldsymbol{\phi}}_\mathrm{SIM} \triangleq \exp(-\jmath\bar{\boldsymbol{\omega}}_\mathrm{SIM})$, and again, $\hat{\mathbf{o}}$ is computed via (15), leading to the following derivations:

$$\frac{\partial \mathfrak{J}}{\partial \bar{\boldsymbol{\omega}}} = \frac{\partial \mathfrak{J}}{\partial \bar{\boldsymbol{\omega}}_\mathrm{SIM}} = \frac{\partial \mathfrak{J}}{\partial \hat{\mathbf{o}}(t)} \cdot \frac{\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}}{\partial \mathbf{y}(t)} \cdot \frac{\partial \mathbf{y}(t)}{\partial \bar{\boldsymbol{\phi}}_\mathrm{SIM}} \cdot \frac{\partial \bar{\boldsymbol{\phi}}_\mathrm{SIM}}{\partial \bar{\boldsymbol{\omega}}_\mathrm{SIM}}, \tag{26}$$

where $\partial\mathfrak{J}/\partial\hat{\mathbf{o}}(t)$ and $\partial f^\mathrm{d}_{\mathbf{w}_\mathrm{d}}/\partial \mathbf{y}(t)$ are the same as before, while, $\partial \bar{\boldsymbol{\omega}}_\mathrm{SIM}/\partial \bar{\boldsymbol{\omega}}_\mathrm{SIM} = -\jmath \exp(-\jmath\bar{\boldsymbol{\omega}}_\mathrm{SIM})$. Since now $\mathcal{T}(\cdot)$ involves the SIM system model of (7), $\partial \mathbf{y}(t)/\partial \bar{\boldsymbol{\phi}}_\mathrm{SIM}$ requires further derivations. Following the same procedure as in (24), and by denoting the response matrix of each of the M SIM elements as $\bar{\boldsymbol{\Phi}}^m_\mathrm{SIM} \triangleq \mathrm{diag}(\bar{\boldsymbol{\phi}}^m_\mathrm{SIM})$, where $\bar{\boldsymbol{\phi}}^m_\mathrm{SIM}$ is the trainable response of the m-th layer it is deduced $\partial \mathbf{y}(t)/\partial \bar{\boldsymbol{\phi}}_\mathrm{SIM} = [[\partial \mathbf{y}(t)/\partial \bar{\boldsymbol{\phi}}^1_\mathrm{SIM}]^\top, \ldots, [\partial \mathbf{y}(t)/\partial \bar{\boldsymbol{\phi}}^M_\mathrm{SIM}]^\top]^\top$ with:

$$\frac{\partial \mathbf{y}(t)}{\partial \bar{\boldsymbol{\phi}}^m_\mathrm{SIM}} = \begin{cases} \begin{aligned} &(\mathbf{s}^\top(t)\mathbf{H}_1^*(t)) \\ &\otimes\Big(\mathbf{H}_2(t) \prod_{m'=M}^{2} \bar{\boldsymbol{\Phi}}^{m'}_\mathrm{SIM}\boldsymbol{\Xi}_{m'}\Big)\mathbf{D}, \end{aligned} & m = 1 \\ \begin{aligned} &\Big(\Big(\prod_{m'=m}^{2} \boldsymbol{\Xi}_{m'}\bar{\boldsymbol{\Phi}}^{m'-1}_\mathrm{SIM}\Big)\mathbf{H}_1^\dagger(t)\mathbf{s}(t)\Big)^\top \\ &\otimes\Big(\mathbf{H}_2(t) \prod_{m'=M}^{m+1} \bar{\boldsymbol{\Phi}}^{m'}_\mathrm{SIM}\boldsymbol{\Xi}_{m'}\Big)\mathbf{D}, \end{aligned} & 1 < m \leq M. \end{cases} \tag{27}$$

B. System Considerations During MINN Deployment

Once our MINN architecture is sufficiently trained, inference may take place on real data. This process can be also largely described by Algorithm 1 with the omission

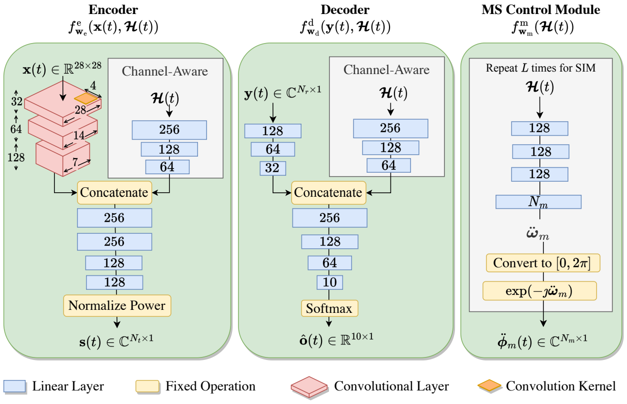
Encoder
$f^{\mathrm{e}}_{\mathbf{w}_{\mathrm{e}}}(\mathbf{x}(t), \boldsymbol{\mathcal{H}}(t))$
$\mathbf{x}(t) \in \mathbb{R}^{28\times 28}$
Channel-Aware
$\boldsymbol{\mathcal{H}}(t)$
4
28
14
7
32
64
128
256
128
64
Concatenate
256
256
128
128
Normalize Power
$\mathbf{s}(t) \in \mathbb{C}^{N_t\times 1}$
Decoder
$f^{\mathrm{d}}_{\mathbf{w}_{\mathrm{d}}}(\mathbf{y}(t), \boldsymbol{\mathcal{H}}(t))$
$\mathbf{y}(t) \in \mathbb{C}^{N_r\times 1}$
128
64
32
Channel-Aware
$\boldsymbol{\mathcal{H}}(t)$
256
128
64
Concatenate
256
128
64
10
Softmax
$\hat{\mathbf{o}}(t) \in \mathbb{R}^{10\times 1}$
MS Control Module
$f^{\mathrm{m}}_{\mathbf{w}_{\mathrm{m}}}(\boldsymbol{\mathcal{H}}(t))$
Repeat L times for SIM
$\boldsymbol{\mathcal{H}}(t)$
128
128
128
N_m
$\ddot{\boldsymbol{\omega}}_m$
Convert to $[0, 2\pi]$
$\exp(-\jmath\ddot{\boldsymbol{\omega}}_m)$
$\ddot{\boldsymbol{\phi}}_m(t) \in \mathbb{C}^{N_m\times 1}$
Linear Layer
Fixed Operation
Convolutional Layer
Convolution Kernel

Fig. 4. DNN implementation of the three modules of the proposed MINN architecture for the considered MNIST classification problem. The channel matrices of $\boldsymbol{\mathcal{H}}(t)$ are flattened to vectors and are concatenated. The channel-aware branches are ignored when channel-agnostic transceivers are used.

of sampling $\mathbf{o}(t)$ at line 4, since it is not available during inference, and, consecutively, the omission of line 10, where the backward pass is performed. In practice, it is reasonable to assume that training takes place at a single network node with sufficient computational and power capabilities, before sharing the obtained neural network weights to the corresponding physical devices once prior the commencement of the deployment stage stage. This alleviates the need for costly signaling procedures and data transfer during training, as well as requirements for sufficient computational power for training by all three network nodes. On the other hand, when each trainable DNN module is physically collocated with its corresponding device, training entails the considerable overhead of exchanging gradient values using a dedicated control channel with sufficient capacity [47], which may result in costly power consumption by the TX, RX, and MS devices. Nevertheless, this methodology provides the benefit of enhanced privacy: The observed input data by the TX need not be transmitted to the RX; the target information is solely required for the gradient computations given in (20).

For the channel-aware transceiver cases, the implied channel estimation, at every time step, should be designed in a way that all physical entities receive the same estimates, which entails tailored orchestration and signaling. When reconfigurable MSs are deployed, their control module should be large enough to perform effective feature extraction and phase configuration control. This dual need for data receiving capabilities and local computation could be addressed by the technology of Hybrid RISs (HRISs) [48]. In fact, by incorporating a power source to feed active components, sensing-capable MSs are able to perform channel estimation locally. If endowed with DNN-optimized computation units, the channel-estimation-phase-configuration process may take place truly autonomously without any additional network overheads, other than the usual pilot exchange messages that need to take place regardless of the type of MS device. The idea of self-configured HRISs to implement MS controllers becomes increasingly attractive when the variation of channel-agnostic transceivers is adopted. Under this choice, the MS is the only network node tasked with channel coding, which takes place over-the-air simultaneously to the inference process with important implications in the hardware requirements for the other devices. The numerical evaluation of the next section provides encouraging results that such a deployment scenario may be facilitated under certain network conditions.

TABLE I
NUMBER OF DNN PARAMETERS FOR DIFFERENT MINNS AND $N_t = 4$.

Method (MS elements)	**Controllable MS & Channel-Aware**	**Fixed MS & Channel-Aware**	**Fixed MS & Channel-Agnostic**
SIM ($3\times 8\times 8$)	2.3×10^6	2.3×10^6	2.5×10^5
SIM ($3\times 12\times 12$)	4.6×10^6	4.6×10^6	2.5×10^5
RIS (16×16)	1.0×10^7	7.7×10^6	2.5×10^5
RIS (25×25)	2.5×10^7	1.8×10^7	2.5×10^5
No Metasurface	9.9×10^5		2.5×10^5
RX-DNN (25×25)	1.3×10^6		

On the opposite direction, a fixed-configuration MS does not need to observe any CSI during operation, which greatly simplifies the system requirements, as well as the overall computational budget. In fact, MSs with fixed phase configurations can be manufactured to be completely passive, essentially reducing a portion of the power consumption of the considered E2E DNN. However, the channel coefficients still need to be available during training to compute the gradient updates, as described in the previous section, which again motivates the use of an HRIS for efficient CSI collection.

VII. NUMERICAL EVALUATION AND DISCUSSION

A. Simulation Setup

To evaluate the proposed MINN framework, we devise a problem of over-the-air MNIST classification [54] under Ricean fading in the presence of an MS in the wireless environment. Advanced DNN architectures [55]–[57] achieve 98%-99.5% classification accuracy, which constitutes an upper bound when "infer-when-transmit" methodology is used. While only $\log_2(10) \approx 3.32$ bits are required for transmission of the class index, the considerable computational cost at the TX (e.g., the top-performing model of [57] contains 1.58×10^8 parameters), motivates this paper's adopted "infer-while-transmitting" paradigm. We particularly consider an N_t-antenna low-cost IoT TX device observing grayscale images of handwritten digits, and wishing to perform GOC with an 32-antenna RX, which intends to obtain the numerical value of the digit of each image. A Cartesian coordinate system with the MS (either an RIS or the first SIM layer) at the origin has been simulated, where the TX and RX were placed at the points $(-2, 2, -0.5)$ and $(10, 16, 4)$, respectively. Narrowband transmissions at 28 GHz with varying Ricean fading conditions were considered [35, Sec. 2, eqs. 2-9]; specifically, we have set the Ricean factors (κ_1, κ_2, and κ_3 in [35]) to 13, 7, and 3 dB for the TX-MS, MS-RX, and TX-RX channels, respectively. In fact, the details of the channel model follow the modeling of [35, Sec. 2], where uniform linear arrays have been used for the TX, RX devices and uniform rectangular arrays for the MSs. It is noted that the TX-RX distance was approximately 19 m, hence, the free-space attenuation of this direct link was at 41.5 dB, while the total attenuation of the MS-enabled E2E link was approximately 67 dB, indicating that, if the MS was to be optimized to enhance the signal strength, only limited gains would be obtained due to the large pathloss difference. For the cases where the MS was an RIS, the surface was oriented parallel to the xz plane to enable reflections, whereas, for the SIM considerations, the cascaded surfaces had their broadside placed parallel to the yz plane to enable diffraction of the signal arriving at their first layer, centered at the origin. Each of the next $M-1$ layers was placed in parallel at distances of $d_M = 5\lambda$ from its predecessor. The centers of the MS elements had a distance of $\lambda/2$ with their neighbors, so that their surface area was $S_M = \lambda^2/4$. Finally, unless otherwise specified, $N_r = 32$, the transmission power was set to $P = 30$ dBm with a noise variance of $\sigma^2 = -90$ dBm, while the value $M = 3$ for the SIM layers was fixed.

B. MINN Versions, Baselines, and Training Parameters

The implementation details of the various simulated DNN modules are included in Fig. 4. Unless otherwise specified, ReLU activation functions were used, and layer-wise normalization was applied in all branches that received $\boldsymbol{\mathcal{H}}(t)$. It is noted that, apart from the layers that are dependent on the system variables (N_m, N_t), the architectures of all DNNs remained fixed for fairness in computational resources, meaning that, for larger parameter values, the networks could be less effective. The numbers of trainable parameters for all considered MINN versions are included in Table I, illustrating the computational requirement increase in channel-aware settings and in case of larger MSs.

The simulated MINN versions were compared against two baselines. The first one was a baseline of the same encoder and decoder modules, but in the absence of any MS; this baseline enables the evaluation of the specific benefits brought by the MS. In fact, this is accommodated by our system model, by setting $\boldsymbol{\Phi}(t) = \mathbf{0}_{N_m \times N_m}$ in (4). The second baseline is a mixture between the “transmit-then-infer” and “DNN splitting” paradigms, as discussed in Section II-C, referred to as “RX-DNN,” where the RX reconstructs the data and employs a DNN classifier trained independently of the channel optimization phase. It considers a conventional communication system, which is designed to encode, modulate, and apply TX/RX beamforming to the image data, upon optimizing the channel’s data rate. The encoder and decoder are the respective modules of a Variational Auto-Encoder (VAE) [21] DNN that is trained independently, and its encoded representations are transmitted over the channel. The details of the system are given in the Appendix.

The MNIST data set contains 7×10^4 images, 10^4 of which were used for testing. Additionally, 10^4 channel realizations were used for training, while 10^3 more were used for testing. At each forward pass, during either training or inference, the input image was paired with a random channel realization. Each training instance took place over 1000 epochs, using the Adam optimizer [53] with learning rate of $\eta = 10^{-4}$ and a weight decay of 10^{-4}. It is observed that this effective expansion of the data set to 3×10^8 image-channel pairs, as well as the stochastic nature of channel realizations, increase the variance of the achieved DNN performance across multiple restarts of the training process. For the experiments, each method was trained 10 times with different initialization seeds.

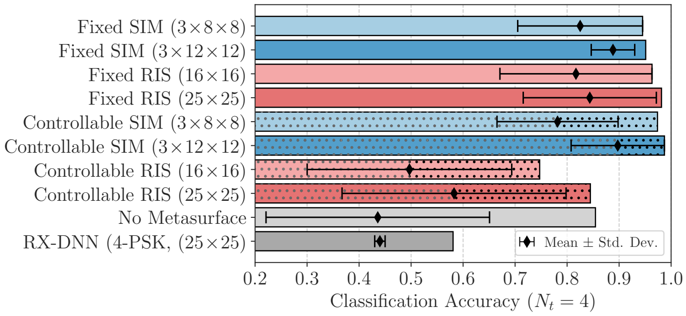

Fig. 5. Comparison of achieved accuracy with different MINN variations and the two adopted baselines, considering $N_t = 4$ and channel-aware transceivers. Bars indicate the highest performance of each method across 10 training restarts, while the black horizontal lines represent the mean accuracy and the standard deviation.

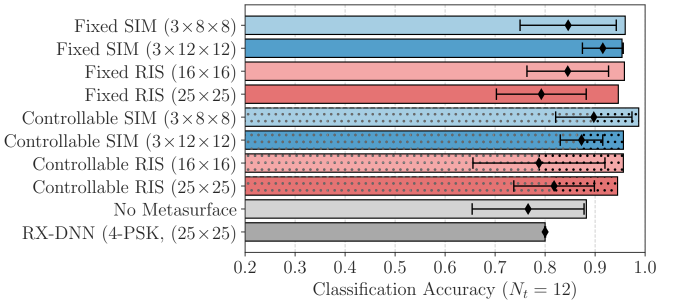

Fig. 6. Achieved accuracy with different MINN variations and the No-MS baseline, considering $N_t = 12$ and channel-aware transceivers.

C. Numerical Results

1) Reconfigurable versus Fixed Metasurfaces: We commence by investigating the benefits of MINNs incorporating reconfigurable MSs (that entail an extra DNN module at the MS controller) in comparison with their trainable fixed-MS-response counterparts (labeled as “Fixed” in the sequel), considering channel-aware transceivers. The test set classification accuracy considering two different RIS and SIM sizes and a TX with $N_t = 4$ is illustrated in Fig. 5. As shown, the top restarts of MINN versions with larger RIS/SIM sizes can achieve near-optimal performance, however, smaller sizes, especially for the cases of RISs, are less effective. Controllable RIS versions result in underwhelming performance due to the apparent inability of the control module to optimize each channel independently using a single MS. It is also depicted that, in the absence of an MS, substantial accuracy is reached. However, the classification task is not solved successfully, since the classification accuracy on MNIST data of even simple DNN classifiers reach scores higher than 0.98 (see also the Appendix). The baseline communication system performs evidently poorly due to difficulties in correctly detecting the transmitted symbols. Few antenna elements and LoS-dominant channels restrict the number of available spatial streams, d, resulting to denser constellations.

Figure 6, where the number of TX antennas is increased to $N_t = 12$, demonstrates that all considered MINN variations

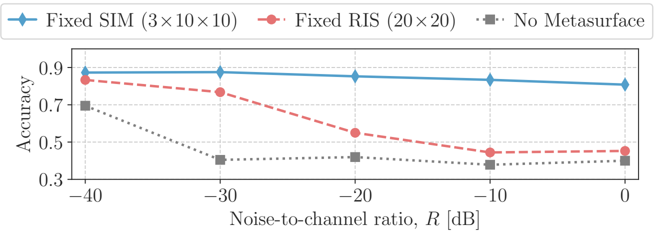

Fig. 7. Mean accuracy with different MINN variations and the No-MS baseline, considering $N_t = 8$ and channel-aware transceivers under noisy channel observations.

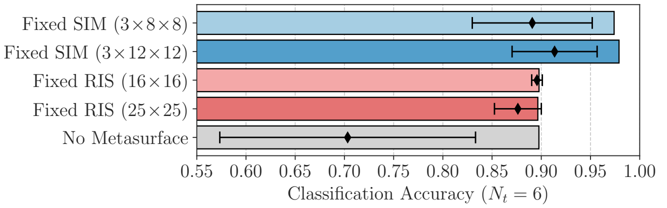

Fig. 8. Achieved accuracy with different MINN variations and the No-MS baseline, considering $N_t = 6$ and channel-agnostic transceivers.

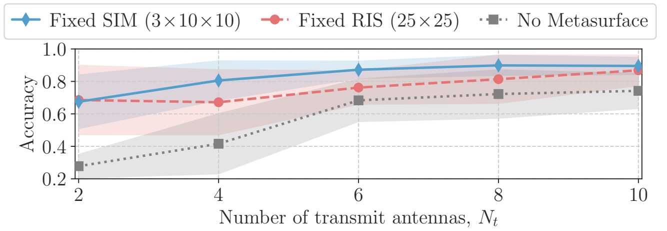

Fig. 9. Mean accuracy with different MINN variations and the No-MS baseline, considering varying numbers of TX antennas, fixed-response MSs, and channel-agnostic transceivers. The shaded regions denote the standard deviations.

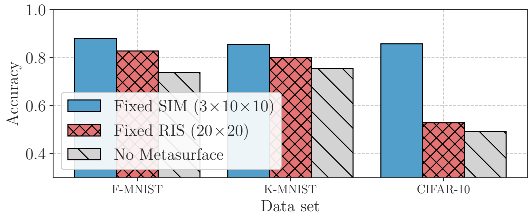

Fig. 10. Mean accuracy with different MINN variations and the No-MS baseline, considering different data sets, fixed-configuration MSs, and channel-agnostic transceivers.

are successful in their GOC task, in contrast to the two baselines. Notice that the variability across trials, indicated by the standard deviation ranges, is greatly reduced compared to the $N_t = 4$ case in Fig. 5, indicating that, as expected, MINN approaches become more robust as the communication resources increase. In addition, by inspecting Figs. 5 and 6, it can be seen that there is a small observed difference in the performance between the controllable and fixed versions of all MINN versions, with the latter exhibiting improved robustness. It can, thus, be concluded that fixed-MS-response approaches provide *more effective E2E DNN architectures* in static wireless systems, in view of the reduced computational, hardware, and system requirements, as discussed in Section VI.

2) Impact of Channel Knowledge and Comparison with Channel-Agnostic Schemes: To investigate the importance of accurate channel knowledge with the channel-aware transceivers, we have conducted experiments with erroneous channel observations. Precisely, for each of the channel matrices in $\boldsymbol{\mathcal{H}}(t)$, let $\mathbf{H}_{\text{true}}(t)$, a noisy estimate of it, $\mathbf{H}_{\text{noisy}}(t)$, was constructed emulating channel estimation errors. Given the amplitude ratio R between $\mathbf{H}_{\text{true}}(t)$ and its injected noise, $\mathbf{H}_{\text{noisy}}(t)$ was generated as $\mathbf{H}_{\text{noisy}}(t) \triangleq \alpha_1(\mathbf{H}_{\text{true}}(t)+\mathbf{N}(t))$, where the intermediate noise $\mathbf{N}(t) \triangleq \alpha_2\mathbf{N}_0(t)$, with $\mathbf{N}_0(t)$ being the random noise matrix following $\mathcal{CN}(\mathbf{0},\mathbf{I})$, and $\alpha_2 \triangleq ||\mathbf{H}_{\text{true}}(t)||_F/(R||\mathbf{N}_0(t)||_F)$. The outer scaling factor $\alpha_1 \triangleq ||\mathbf{H}_{\text{true}}(t)||_F/||\mathbf{H}_{\text{true}}(t) + \mathbf{N}(t)||_F$ ensures proper normalization, yielding $||\mathbf{H}_{\text{true}}(t)||_F = ||\mathbf{H}_{\text{noisy}}(t)||_F$. For this scenario, the DNNs observed $\mathbf{H}_{\text{noisy}}(t)$ instances for all input channels, while transmissions took place with $\mathbf{H}_{\text{true}}(t)$ in $\mathcal{T}(\cdot)$. The mean achieved accuracies of investigated MINNs under increasing R values are shown in Fig. 7. It is shown that SIM exhibits a small, yet noticeable, performance degradation at larger noise levels, while the RIS-based MINN and the No-MS baseline are severely impacted, which further exemplifies the robustness achieved by the multi-layer OAC processing.

On the other hand, MINN variations with fixed-MS-responses, channel-agnostic transceivers, and $N_t = 6$ are compared in Fig. 8. It can be observed that both SIM-based architectures may achieve *near optimal performance without the need of CSI acquisition* during network deployment, which has important benefits in the simplification of the overall GOC system design on top of exhibiting more stable training behavior across the training restarts, as it was shown in Figs 5 and 6. It is noted that the absence of CSI leads to smaller E2E DNN sizes, due to the exclusion of the channel coding branches whose parameters grow linearly with the number of channel coefficients. This reduction in DNN size correlates with the reduced variability of the methods' performances across re-runs, which hints that much more extended training periods would be required for effective training the channel-aware architectures of the previous section. The performance of channel-agnostic MINNs with fixed MS configurations is further evaluated in Fig. 9 across varying TX antenna values. Similar to the behavior of Figs. 5 and 6, the accuracy increases with higher spatial diversities offered by increasing the values of N_t, while the variance across restarts diminishes, with the No-MS baseline being particularly inefficient in smaller antenna sizes.

3) Other Data Sets: We have also investigated whether the observed performance benefits in terms of accuracy are maintained under three more challenging data sets [54]. The "Fashion MNIST" (F-MNIST) and the "Kuzushiji MNIST" (K-MNIST) variations of the original MNIST have been selected that provide more intricate objects for image classification using the shape image dimensions number of classes as the MNIST. Therefore, the same architectures as presented in Fig. 4 were kept. As upper bounds, we have trained DNN instances of three convolutional and three linear layers that achieved 0.93 and 0.91 accuracy, respectively. The CIFAR-10 data set has been also considered, containing RGB photographs of 32 × 32 pixels, which presents a demanding

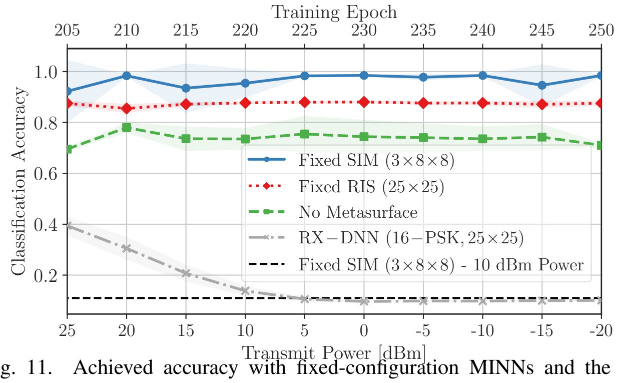

Fig. 11. Achieved accuracy with fixed-configuration MINNs and the two adopted baselines, considering channel-agnostic transceivers, for progressive TX power decrement over training.

classification problem. Although the goal of this work is not to provide top performing DNN architectures, we adopted deeper network architectures: a Convolutional Neural Network (CNN) of eight convolutional layers with 3×3 kernels and increasingly larger channel dimensions up to 512 with in-between max-pooling operations, followed by four wide linear layers of 4096 units. ReLU activations, dropout, and batch normalization techniques have been used in addition to typical preprocessing transformations for data augmentation. This architecture was trained to achieve 0.93 classification accuracy.

To implement the corresponding MINN, a similar DNN architecture was adopted as the encoder, while dropping the last three CNN and all original linear layers to introduce a single linear layer tasked to transform the outputs of the last convolutional layer to the appropriate dimension for transmission. The decoder was kept as in Fig. 4. Considering that channel-agnostic versions provide more stable training performance, we did not incorporate channel inputs to the transceiver nodes, and only fixed-MS versions were realized due to the prolonged training periods. Critically, the extension of the system model discussed in Section V-D1 has been implemented to take advantage of the temporal degree of freedom. We have assumed that $\tau = 4$ transmissions take place within a fixed realization, hence, the encoder outputted a complex vector of dimension τN_t, the data of which was transmitted sequentially. The RX collected and concatenated the τ received vectors before performing its forward pass.

The performance of the considered MINNs for the latter data sets is depicted in Fig. 10, where the mean accuracy over three restarts is reported. As shown, the MNIST variants exhibit similar trends as the original data set, where the SIM-based MINN outperforms the RIS case, which in turn has an edge over the No-MS baseline. The differences of the methods are particularly pronounced under the challenging CIFAR-10 data set, where the RIS-based MINN and the baseline are unable to achieve satisfactory performance. Indicatively, the mean accuracy of the SIM-based MINN amounts to 91%, 95%, and 95% of the achieved accuracies of the upper bound DNN counterparts for F-MNIST, K-MNIST, and CIFAR-10, trained without channel interferences.

4) Transmission and Computation Power: Trained DNNs are expected to be robust to increased noise levels in the inference data sets without or with minimal extra training. To investigate this hypothesis, we have implemented the following procedure for two MINN versions with fixed-MS-responses, considering channel-agnostic transceivers. Once the first 200 training epochs elapsed, we continued the training process by decreasing the transmission power by 5 dBm every five epochs, starting from $P = 30$ dBm until we reach -20 dBm. In this way, the first epochs were used as pre-training, while the latter ones fine-tuned the model to the changing conditions. The motivation for this approach is that DNNs typically perform most of their learning during the initial iterations, while few final iterations only refine the learning process by imposing small changes to the learned weights. This practice can be regarded as a implementation of transfer learning [58], where DNNs are fine-tuned to adapt to different data regimes. Recall that, for in Figs. 5–8, $P = 30$ dBm was used throughout the entire training process to provide high SNR conditions.

As illustrated in Fig. 11 for the MNIST data set, all considered MINN approaches, as well as the baseline scheme without an MS, exhibit stable performance despite the power drop. It can be seen that the fixed-SIM-response MINN approach provides the best classification accuracy, indicating that once trained, it can successfully solve the inference task *even at 50 dB of lower SNR levels*, which offers substantial energy benefits in the deployment of such GOC approaches. As a baseline, training the same MINN architectures as in Fig. 11 while keeping a fixed transmission power of 10 dBm throughout the learning period resulted to a mean accuracy of 0.11, which showcases the importance of high-SNR data at the initial stages of MINN optimization. This is inline with results from transfer learning [58], and especially previous works that have used similar methodology to adapt DNNs to low-SNR regimes without performance degradation [59]. As expected, the performance of the traditional communication system is heavily impacted by the decrease of the transmission power.

Finally, we henceforth briefly address the computational power consumption. It is noted, however, that a thorough analysis relies on the specific details of the employed hardware and MS implementation which lies well-beyond the scope of this paper. We estimate the energy consumption at the TX per data instance as $E \triangleq P_{\text{net}}\tau_{\text{inf}}/|\mathcal{D}|$, where $P_{\text{net}} \triangleq P_{\text{GPU}} + P_{\text{CPU}}$ with P_{GPU} capturing the power consumption of the Graphics Processing Unit (GPU) that runs the DNN, P_{CPU} includes all other computations for data processing, encoding, and modulation, whereas τ_{inf} is the processing latency in seconds to complete inference over the testing data set. The corresponding values, as shown in Table II, have been obtained through measurements on an NVIDIA RTX 3060 workstation used for the simulations, disregarding the power at the simulated RF-frontend device. A MINN with channel-agnostic transceivers has been considered for the "infer-while-transmitting" case, while the previously discussed fully digital DNN and RX-DNN methodologies are included in the other two strategies. Clearly, the energy consumption of the MINN strategy is comparable to the lightweight RX-DNN (achieving much lower accuracy), while its benefits over "infer-then-transmit" are exemplified in the harder CIFAR-10 case.

TABLE II
ESTIMATED COMPUTATIONAL ENERGY CONSUMPTION FOR THE TX DEVICE UNDER THE THREE EI PARADIGMS.

MNIST			
Strategy	P_{net}(W)	τ_{inf}(s)	E(mJ/inst.)
Infer-then-transmit	41.97	8.55	5.98
Transmit-then-infer	44.43	3.27	2.42
Infer-while-transmitting (MINN)	39.61	3.62	2.39
CIFAR-10			
Strategy	P_{net}(W)	τ_{inf}(s)	E(mJ/inst.)
Infer-then-transmit	89.47	10.63	15.85
Transmit-then-infer	43.04	3.68	2.64
Infer-while-transmitting (MINN)	43.82	5.30	3.87

VIII. CONCLUSIONS

In this paper, we proposed the framework of MINNs that enables EI by treating the MS-programmable wireless channel as a hidden OAC artificial neural network layer(s), in sharp contrast to previous literature that considers the signal propagation environment as a source of noise. Architectural MINN variations have been presented that consider RISs and SIM, either controllable via dedicated DNN modules or with trainable fixed response configurations, while both channel-aware and channel-agnostic transceivers have been considered. A variation of the backpropagation algorithm for fading channels has been developed, and system considerations that concern data collection, channel acquisition, and hardware requirements have been discussed. Our performance evaluations highlighted that MINNs are more efficient classification schemes compared to MS-free or traditional "transmit-then-compute" systems, with the same link budget measured in terms of TX power, number of antennas and MS elements. The key insights of our numerical analysis showcased that, for the considered system parameters, fixed MS responses are more effective E2E DNN architectures in view of the reduced computations, hardware, and system requirements, while CSI knowledge is not a hard requirement for the transceiver, since channel coding may take place over-the-air via SIM. Finally, it was demonstrated that, once pre-trained under high SNR, fixed-MS-response MINN architectures can be sufficiently robust to a wide range of SNR conditions, offering a strong comparative advantage for practical EI deployment.

APPENDIX
BASELINE COMMUNICATION SYSTEM

In this section, we design a MIMO communication system for MNIST classification with independent source and channel coding. For ease of notation, we drop index t, noting that all operations take place at every channel frame. First, a VAE [21] is used as a data-driven lossy compression scheme, whose operation is defined as $\hat{\mathbf{x}} \triangleq f_{\text{dec}}^{\text{VAE}}(f_{\text{enc}}^{\text{VAE}}(\mathbf{x}))$, where the encoder (that is placed at the TX) generates a latent space representation $\boldsymbol{\xi} \triangleq f_{\text{enc}}^{\text{VAE}}(\mathbf{x}) \in \mathbb{R}^{2\times 1}$, while the decoder (at the RX side) learns to reconstruct the original input $\mathbf{x}$ as $\hat{\mathbf{x}} = f_{\text{dec}}^{\text{VAE}}(\boldsymbol{\xi})$. The DNN architecture for VAE is kept as in [21], achieving $O(10^{-4})$ MSE reconstruction error once pretrained on MNIST data. The two elements of $\boldsymbol{\xi}$, treated as two 32-bit floating point numbers, are then modulated via K-ary Phase Shift Keying (PSK), where K is determined by evaluating the performance of this method for different K values for each scenario to obtain the number of transmitted symbols $d = \lceil 64/\log_2 K \rceil \leq \min\{N_t, N_r\}$, which are included in the symbol $\mathbf{z} \in \mathbb{C}^{d\times 1}$, as shown in Fig. 12.

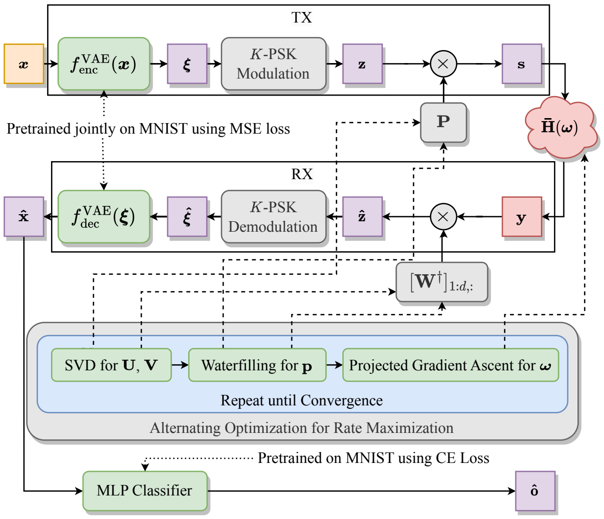
TX
$\boldsymbol{x}$
$f_{\text{enc}}^{\text{VAE}}(\boldsymbol{x})$
$\boldsymbol{\xi}$
K-PSK Modulation
$\mathbf{z}$
$\mathbf{s}$
$\mathbf{P}$
$\bar{\mathbf{H}}(\boldsymbol{\omega})$
Pretrained jointly on MNIST using MSE loss
RX
$\hat{\mathbf{x}}$
$f_{\text{dec}}^{\text{VAE}}(\boldsymbol{\xi})$
$\hat{\boldsymbol{\xi}}$
K-PSK Demodulation
$\hat{\mathbf{z}}$
$\mathbf{y}$
$[\mathbf{W}^{\dagger}]_{1:d,:}$
SVD for $\mathbf{U}$, $\mathbf{V}$
Waterfilling for $\mathbf{p}$
Projected Gradient Ascent for $\boldsymbol{\omega}$
Repeat until Convergence
Alternating Optimization for Rate Maximization
Pretrained on MNIST using CE Loss
MLP Classifier
$\hat{\mathbf{o}}$

Fig. 12. Block diagram of a conventional MIMO communication system for MNIST classification with independent source and channel coding.

By denoting the RIS-aided E2E channel as $\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}}) \triangleq \mathbf{H}_{\text{D}} + \mathbf{H}_2 \text{diag}(\exp(-\jmath\dot{\boldsymbol{\omega}}))\mathbf{H}_1^{\dagger}$ and its Singular Value Decomposition (SVD) as $\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}}) = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\dagger}$, $\mathbf{P} \triangleq \text{diag}(\sqrt{\mathbf{p}})[\mathbf{V}]_{:,1:d} \in \mathbb{C}^{N_t \times d}$ can be used as the precoding matrix with $\mathbf{p}$ including the per-symbol power allocation, i.e., the transmitted signal is $\mathbf{s} = \mathbf{P}\mathbf{z}$. At the RX, weighted minimum mean squared error combining is adopted, yielding the estimation $\hat{\mathbf{z}} = [\mathbf{W}^{\dagger}]_{1:d,:}\mathbf{y}$ with $\mathbf{W} \triangleq \left(\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}})\mathbf{P}\mathbf{P}^{\dagger}\bar{\mathbf{H}}^{\dagger}(\dot{\boldsymbol{\omega}}) + \sigma^2\mathbf{I}\right)^{-1}\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}})\mathbf{P}$. Notations $[\mathbf{V}]_{:,1:d}$ and $[\mathbf{W}^{\dagger}]_{1:d,:}$ represent respectively the first d columns of $\mathbf{V}$ and the first d rows of $\mathbf{W}^{\dagger}$. Then, the rate maximizing power allocation $\mathbf{p}$ and the RIS phase configuration $\dot{\boldsymbol{\omega}}$ can be obtained from the solution of the optimization problem:

$$\mathcal{OP}_1 : \max_{\mathbf{p},\dot{\boldsymbol{\omega}}} \sum_{i=1}^{\min\{N_t,N_r\}} \log_2\left(1 + \frac{[\mathbf{p}]_i\tilde{\sigma}_i(\dot{\boldsymbol{\omega}})}{\sigma^2}\right)$$
$$\text{s.t. } \|\mathbf{p}\| = P,\ \dot{\boldsymbol{\omega}} \in [0, 2\pi)^{N_m},$$

where $\tilde{\sigma}_i(\dot{\boldsymbol{\omega}})$ is the n-th singular value of $\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}})$. The latter approach can be followed in an alternating optimization manner to design the TX precoder, RX combiner, and RIS phase configuration maximizing the achievable spectral efficiency. In particular, we iterate until convergence between: Step 1) for a given θ, obtain $\mathbf{p}$ from the waterfilling solution of $\mathcal{OP}_1$ as well as $\mathbf{P}$ and $\mathbf{W}$ using also $\bar{\mathbf{H}}(\dot{\boldsymbol{\omega}})$'s SVD; and Step 2) for previous step's $\mathbf{p}$ as well as $\mathbf{V}$ and $\mathbf{U}$ from the previous iteration, perform projected gradient ascent using automatic differentiation to find $\dot{\boldsymbol{\omega}}$ solving $\mathcal{OP}_1$.

Once $\hat{\mathbf{z}}$ is retrieved, it is demodulated to obtain $\hat{\boldsymbol{\xi}}$, which is then fed into $f_{\text{dec}}^{\text{VAE}}(\hat{\xi})$ to obtain the reconstructed image $\hat{\mathbf{x}}$. A CNN classifier, complementing our baseline communication system summarized in Fig. 12, has been pretrained on original MNIST images (i.e., instances of $\mathbf{x}$) with over 0.98 classification accuracy. In our experimentation, we have evaluated CNN on unseen instances of $\hat{\mathbf{x}}$ to assess this baseline's classification accuracy.

References

[1] E. Calvanese Strinati *et al.*, "Towards distributed and intelligent integrated sensing and communications for 6G networks," *IEEE Wireless Commun.*, vol. 32, no. 1, pp. 60–67, 2025.

[2] M. Uusitalo *et al.*, "European vision for the 6G network ecosystem," Sep. 2024. [Online]. Available: https://doi.org/10.5281/zenodo.13708425

[3] P. Di Lorenzo *et al.*, "Goal-oriented communications for the IoT: System design and adaptive resource optimization," *IEEE Internet Things Mag.*, vol. 6, no. 4, pp. 26–32, 2023.

[4] E. Calvanese Strinati *et al.*, "Goal-oriented and semantic communication in 6G AI-native networks: The 6G-GOALS approach," in *Proc. Joint Europ. Conf. Netw. Commun. & 6G Summit*, Antwerp, Belgium, 2024.

[5] M. Merluzzi *et al.*, "Wireless edge machine learning: Resource allocation and trade-offs," *IEEE Access*, vol. 9, pp. 45 377–45 398, 2021.

[6] F. Pezone *et al.*, "Goal-oriented communication for edge learning based on the information bottleneck," *arXiv preprint arXiv:2202.12639*, 2022.

[7] H. Ye *et al.*, "Deep over-the-air computation," in *Proc. IEEE Int. Conf. Commun.*, virtual, 2020.

[8] H. Xie *et al.*, "Deep learning enabled semantic communication systems," *IEEE Trans. Signal Process.*, vol. 69, pp. 2663–2675, 2021.

[9] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 1948.

[10] M. Jankowski *et al.*, "Wireless image retrieval at the edge," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 89–100, 2021.

[11] E. Basar *et al.*, "Reconfigurable intelligent surfaces for 6G: Emerging hardware architectures, applications, and open challenges," *IEEE Veh. Technol. Mag.*, vol. 19, no. 3, pp. 27–47, 2024.

[12] J. An *et al.*, "Stacked intelligent metasurfaces for efficient holographic MIMO communications in 6G," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 8, pp. 2380–2396, 2023.

[13] G. Huang *et al.*, "Stacked intelligent metasurfaces for task-oriented semantic communications," *arXiv preprint arXiv:2407.15053*, 2024.

[14] N. Hello *et al.*, "Optimizing RIS impairments through semantic communication," *arXiv preprint arXiv:410.08155*, 2024.

[15] X. Lin *et al.*, "All-optical machine learning using diffractive deep neural networks," *Science*, vol. 361, no. 6406, pp. 1004–1008, 2018.

[16] C. Liu *et al.*, "A programmable diffractive deep neural network based on a digital-coding metasurface array," *Nat. Electron.*, vol. 5, no. 2, pp. 113–122, 2022.

[17] S. Chen *et al.*, "RIS-based on-the-air semantic communications - A diffractional deep neural network approach," *IEEE Wireless Commun.*, vol. 31, no. 4, 2024.

[18] I. Goodfellow *et al.*, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[19] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, no. 4, pp. 303–314, 1989.

[20] F. Binucci *et al.*, "Enabling edge artificial intelligence via goal-oriented deep neural network splitting," in *Proc. Int. Symp. Wireless Commun. Syst.*, Rio de Janeiro, Brazil, 2024.

[21] D. P. Kingma *et al.*, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, Banff, AB, Canada, 2014.

[22] O. Ronneberger *et al.*, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Med. Image Comput. Comput. Assist. Interv.*, Munich, Germany, 2015.

[23] P. A. Stavrou and M. Kountouris, "A rate distortion approach to goal-oriented communication," in *Proc. IEEE Int. Symp. Inf. Theory*, Espoo, Finland, 2022.

[24] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, p. 1929–1958, 2014.

[25] C. Gulcehre *et al.*, "Noisy activation functions," in *Proc. Int. Conf. Mach. Learn.*, vol. 48, New York, USA, 2016.

[26] D. P. Kingma *et al.*, "Variational dropout and the local reparameterization trick," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, Montreal, Canada, 2015.

[27] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 48, New York, USA, 2016.

[28] Z. Wang *et al.*, "Over-the-air computation for 6G: Foundations, technologies, and applications," *IEEE Internet Things J.*, vol. 11, no. 14, pp. 24 634–24 658, 2024.

[29] S. Dörner *et al.*, "Deep learning based communication over the air," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 132–143, 2018.

[30] E. Bourtsoulatze *et al.*, "Deep joint source-channel coding for wireless image transmission," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Brighton, UK, 2019.

[31] D. Gündüz *et al.*, "Beyond transmitting bits: Context, semantics, and task-oriented communications," *IEEE J. Select. Areas Commun.*, vol. 41, no. 1, pp. 5–41, 2023.

[32] B. Nazer and M. Gastpar, "Computation over multiple-access channels," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3498–3516, 2007.

[33] B. Xiao *et al.*, "Over-the-air federated learning: Status quo, open challenges, and future directions," *Fundam. Res.*, 2024.

[34] G. C. Alexandropoulos *et al.*, "Phase configuration learning in wireless networks with multiple reconfigurable intelligent surfaces," in *Proc. IEEE Global Commun. Conf.*, 2020.

[35] ——, "Pervasive machine learning for smart radio environments enabled by reconfigurable intelligent surfaces," *Proc. IEEE*, vol. 110, no. 9, pp. 1494–1525, 2022.

[36] G. Stamatelis *et al.*, "Evolving multi-branch attention convolutional neural networks for online RIS configuration," *IEEE Trans. Cognitive Commun. Netw.*, early access, 2025.

[37] H. Liu *et al.*, "Multi-user MISO with stacked intelligent metasurfaces: A DRL-based sum-rate optimization approach," *arXiv preprint arXiv:2408.04837*, 2024.

[38] C. Qian *et al.*, "Dynamic recognition and mirage using neuro-metamaterials," *Nat. Commun.*, vol. 13, no. 1, p. 2694, May 2022.

[39] J. An *et al.*, "Emerging technologies in intelligent metasurfaces: Shaping the future of wireless communications," *IEEE Trans. Antennas Propag.*, early access, 2025.

[40] Q. Ma *et al.*, "Information metasurfaces and intelligent metasurfaces," *Photon. Insights*, vol. 1, no. 1, p. R01, Aug 2022.

[41] X. Luo *et al.*, "Metasurface-enabled on-chip multiplexed diffractive neural networks in the visible," *Light Sci. Appl.*, vol. 11, no. 1, p. 158, May 2022.

[42] K. R. R. Ranasinghe *et al.*, "A doubly-dispersive MIMO channel model parametrized with stacked intelligent metasurfaces," *arXiv preprint arXiv:2501.07724*, 2025.

[43] B. Yang *et al.*, "Reconfigurable intelligent computational surfaces: When wave propagation control meets computing," *IEEE Wireless Commun.*, vol. 30, no. 3, pp. 120–128, 2023.

[44] Z. R. Omam *et al.*, "Holographic metasurfaces enabling wave computing for 6G: Status overview, challenges, and future research trends," *arXiv preprint arXiv:2501.05173*, 2025.

[45] X. Zhang *et al.*, "Reconfigurable intelligent computational surfaces for MEC-assisted autonomous driving networks: Design optimization and analysis," *arXiv preprint arXiv:2407.00933*, 2024.

[46] D. Gündüz *et al.*, "Joint source–channel coding: Fundamentals and recent progress in practical designs," *Proc. IEEE*, pp. 1–32, early access, 2024.

[47] F. Saggese *et al.*, "On the impact of control signaling in RIS-empowered wireless communications," *IEEE Open J. Commun. Society*, vol. 5, pp. 4383–4399, 2024.

[48] G. C. Alexandropoulos *et al.*, "Hybrid reconfigurable intelligent metasurfaces: Enabling simultaneous tunable reflections and sensing for 6G wireless communications," *IEEE Veh. Technol. Mag.*, vol. 19, no. 1, pp. 75–84, 2024.

[49] P. Gavriilidis *et al.*, "Active reconfigurable intelligent surfaces: Circuit modeling and reflection amplification optimization," *IEEE Open J. Commun. Soc.*, vol. 6, pp. 5693–5711, 2025.

[50] N. Kolomvakis and E. Björnson, "Nonlinear distortion issues created by active reconfigurable intelligent surfaces," in *Proc. EuCAP*, Glasgow, UK, 2024.

[51] K. Stylianopoulos and G. C. Alexandropoulos, "Universal approximation with XL MIMO systems: OTA classification via trainable analog combining," *arXiv preprint arXiv:2504.12758*, 2025.

[52] V. S. Borkar, *Basic Convergence Analysis*. Gurgaon: Hindustan Book Agency, 2008, pp. 10–20.

[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, San Diego, USA, 2015.

[54] Torch Contributors, "torchvision.datasets," *pytorch.org*, 2025 [Online], docs.pytorch.org/vision/main/datasets.html. [Accessed: Oct. 11, 2025].

[55] L. Wan *et al.*, "Regularization of neural networks using dropconnect," in *Proc. ICML*, 2013, p. III–1058–III–1066.

[56] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. NeurIPS*, 2017, p. 3859–3869.

[57] A. Tang *et al.*, "FusionBench: A comprehensive benchmark of deep model fusion," *arXiv preprint arXiv:2406.03280*, 2024.

[58] C. Tan *et al.*, "A survey on deep transfer learning," in *Proc. Artif. Neural Netw. Mach. Learn.*, Rhodes, Greece, 2018, pp. 270–279.

[59] Z. Ji *et al.*, "Transfer learning guided noise reduction for automatic modulation classification," *arXiv preprint arXiv:411.08376*, 2024.