EM DeepRay: An Expedient, Generalizable, and Realistic Data-Driven Indoor Propagation Model

Stefanos Bakirtzis, *Member, IEEE*, Jiming Chen, Kehai Qiu, *Student Member, IEEE*, Jie Zhang, *Senior Member, IEEE*, and Ian Wassell

***Abstract*— Efficient and realistic indoor radio propagation modeling tools are inextricably intertwined with the design and operation of next-generation wireless networks. Machine-learning (ML)-based radio propagation models can be trained with simulated or real-world data to provide accurate estimates of wireless channel characteristics in a computationally efficient way. However, most of the existing research works on the ML-based propagation models focus on outdoor propagation modeling, while indoor data-driven propagation models remain site-specific with limited scalability. In this article, we present an efficient and credible ML-based radio propagation modeling framework for indoor environments. Specifically, we demonstrate how a convolutional encoder–decoder can be trained to replicate the results of a ray tracer, by encoding physics-based information of an indoor environment, such as the permittivity of the walls, and decoding it as the path loss (PL) heatmap for an environment of interest. Our model is trained over multiple indoor geometries and frequency bands, and it can eventually predict the PL for unknown indoor geometries and frequency bands within a few milliseconds. In addition, we illustrate how the concept of transfer learning can be leveraged to calibrate our model by adjusting its preestimate weights, allowing it to make predictions that are consistent with measurement data.**

***Index Terms*— 5G, deep learning, indoor radio communication, machine learning (ML), radio propagation, ray tracing.**

I. Introduction

DURING the past decades, wireless communication systems have experienced immense growth and they are now an indispensable part of our every day life. With the advent of the Internet of Things (IoT), and the establishment of 5G and beyond systems, the landscape in the wireless ecosystem is expected to change radically [1]–[3]. Unlike legacy communication systems, next-generation wireless networks are anticipated to host a vast number of technologies and applications with diverse requirements. Hence, efficient network planning and optimization is becoming a much more convoluted and manifold problem, rendering conventional communication network design techniques obsolete [4].

Artificial intelligence is a key enabler toward the reshaping of wireless networks and the shifting from a reactive to a proactive network design approach [5]. In particular, the potential of machine learning (ML) algorithms, a subfield of artificial intelligence, to process large volumes of data and extract purposeful information, can revolutionize wireless network operation [6]. Until now, ML has been used to tackle a wide range of wireless-network-design-related problems, such as resource allocation, user mobility analysis, localization, and wireless channel modeling [7]–[9]. The latter case has recently attracted significant interest, as radio propagation modeling is the cornerstone of the cellular network design [10], [11].

Conventionally, for radio channel modeling, empirical or deterministic models are used. The empirical channel models, such as the COST-231 model [12], are derived by fitting simplistic mathematical models to measurement campaign data [13], [14]. Although these models are practical in use, they can demonstrate significant deviations from the actual received path loss (PL) values [15], potentially rendering them unreliable. Deterministic models rely on the governing laws of electromagnetic wave propagation, providing an approximate solution to Maxwell's equations. The commonly used deterministic models include ray tracing, the finite-difference time-domain method, or the vector parabolic equation method [16]. Unlike empirical models, deterministic models are site-specific, solving Maxwell's equations within a specified physical geometry, thereby yielding more accurate results.

For that reason, the popularity of deterministic models has grown constantly over recent years. Among the various deterministic models, ray tracing has been widely used to calculate radio channels characteristics, and it is expected to have a leading role in the deployment and the design of 5G and beyond systems [17]. A key difference between 5G and legacy communication system design is that a significant part of 5G radio access networks will be installed in indoor

Manuscript received December 8, 2021; revised March 22, 2022; accepted April 17, 2022. Date of publication May 9, 2022; date of current version June 13, 2022. This work was supported by the European Commission through the Horizon 2020 Framework Program, H2020-MSCA-ITN-2019, MSCA-ITN-EID, under Grant 860239, BANYAN. *(Corresponding author: Stefanos Bakirtzis.)*

Stefanos Bakirtzis and Kehai Qiu are with the Department of Computer Science and Technology, University of Cambridge, Cambridge CB3 0FD, U.K., and also with Ranplan Wireless Network Design Ltd., Cambridge CB23 3UY, U.K. (e-mail: stefanos.bakirtzis@ranplanwireless.com; stefanos.bakirtzis@gmail.com; ssb45.cam.ac.uk; kq218@cam.ac.uk).

Jiming Chen is with Ranplan Wireless Network Design Ltd., Cambridge CB23 3UY, U.K. (e-mail: jiming.chen@ranplanwireless.com).

Jie Zhang is with the Department of Electronic and Electrical Engineering, The University of Sheffield, Sheffield S10 2TN, U.K., and also with Ranplan Wireless Network Design Ltd., Cambridge CB23 3UY, U.K. (e-mail: jie.zhang@sheffield.ac.uk).

Ian Wassell is with the Department of Computer Science and Technology, University of Cambridge, Cambridge CB3 0FD, U.K. (e-mail: ijw24@cam.ac.uk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAP.2022.3172221.

Digital Object Identifier 10.1109/TAP.2022.3172221

environments. Traditionally, in-building traffic is served by outdoor cells, following an outside–in approach, and until now only a small number of buildings have dedicated indoor mobile networks. However, in the 5G and beyond era, the majority of in-building mobile traffic will be served by indoor base stations or access points [18]. Thus, an accurate and expedient indoor propagation modeling tool is now more important than ever.

Computational efficiency usually appears as a bottleneck for ray tracing, due to the substantial simulation time and memory required to trace all the ray paths, when the number of scattering objects and ray intersections within the simulated space increases. Data-driven approaches aim to alleviate this limitation by integrating ray tracing simulators with ML algorithms which are capable of learning and inferring radio propagation parameters [10], [11]. In particular, artificial neural networks (ANNs) have been widely used in an effort to expedite [19] or even replace ray tracing simulators [20]. The preponderance of past research concentrates on urban propagation scenarios [20]–[27]; however, as has been mentioned, indoor propagation modeling is of high significance in the deployment of 5G networks. Currently, most of the existing approaches on indoor propagation modeling are confined to the use of simple multilayer perceptrons (MLPs) [28] to determine the radio channel characteristics [19]. This poses limitations in terms of model's generalizability, as MLPs trained within a certain geometry are agnostic to the characteristics of other geometries (materials, building layout, transmitter position) [29]. In addition, it is necessary to increase the fidelity of ML-based propagation models by considering real measured data, instead of using simulated data only [22], [29], [30].

The main contribution of this article is to tackle these two fundamental limitations of indoor ML-based propagation models. To that end, first we outline how a convolutional encoder–decoder can be trained to predict the PL for an arbitrarily complex indoor environment over multiple frequency bands. In particular, we use a modified version of the U-Net architecture using stacked dilated convolutions (SDU-Net) [31], which learns to transform an input tensor, comprising information regarding the physical properties of an indoor environment, to a PL heatmap. Unlike [19], [29], and [30], the predictions of the proposed data-driven framework are based on the electromagnetic properties of the physical environment, such as the permittivity and the conductivity of the walls, and other parameters that affect wave propagation, e.g., the distance and the frequency. We demonstrate that our approach overcomes the problem of limited generalizability, as it is directly applicable to multiple indoor geometries and frequencies, without any further training. The predictions made by our model can replicate closely the results of a ray tracing simulator, with the additional advantage of a substantially reduced computational time (less than a second in a GPU).

To tackle the second limitation, in this article we present an approach that allows making predictions that match measured data. Thence, we use the idea of transfer learning to calibrate the proposed data-driven propagation model. Specifically, we introduce an expedient method that allows our model to adjust its preestimated weights, computed using simulated data, and provide realistic estimates of the signal level using only a very small quantity of measured data. To summarize, in this article we show how to leverage the results from a ray tracing simulator to create a generalizable data-driven propagation model for indoor environments. Once trained, our model can be used as a standalone propagation solver to predict the PL for unknown indoor geometries and frequencies. Consequently, it can be readily calibrated to provide realistic estimates of the signal level for any indoor environment, using only a few measurements.

Specifically, our work differs from previous research in the following ways.

1) While urban and suburban ML-propagation models have been widely explored [20]–[26], the ML-based indoor propagation models have received less attention, despite their importance for 5G and beyond systems. In our work, we introduce a standalone indoor radio propagation model, which builds on input features that consider the particularities of wave propagation in indoor environments. For instance, in indoor environments it is common to encounter a larger variety of building materials than in outdoor spaces. Moreover, in outdoor ray tracing simulations the building walls are modeled as semi-infinite spaces because of the high losses. However, the walls found in indoor environments are typically thinner, and for materials such as plasterboard or wood, the losses are considerably smaller. Hence, it is necessary to consider waves' attenuation when propagating through walls.
2) Instead of using conventional convolutional neural networks (CNNs) and/or MLPs [19], [23], [25], [26], [30], we leverage the computational efficiency of convolutional encoders–decoders to create a generalizable data-driven propagation model. Unlike [20] and [24], the convolutional encoder–decoder used in this article uses the concept of atrous convolutions, which as was shown in [29] and [31] can be considerably advantageous.
3) Our approach is directly applicable to new indoor environments and frequency bands, without further training, allowing PL estimation in a few milliseconds. On the contrary, the predictions of MLP-based models using relative coordinates, environmental features, and other tabular information [19], [25] are restricted to the environment at which the MLP is trained. Likely, the data-driven models in [23] and [24] were trained over a sole operating frequency, or in [24] and [26] the testing was performed on parts of cities included in the training set.
4) We do not confine the comparison of our model's performance only with synthetic ray tracing data 22]–[24], [30]. We also demonstrate that our model can provide estimates of the signal strength, for unknown geometries and frequencies, which are in close agreement with real-world measurements. More importantly, we outline how sparse measurements can be leveraged through transfer learning to make realistic predictions of the signal strength, minimizing the error between our model's predictions and some measured data.

The outline of this article is as follows. First, in Section II, we discuss the functionality of some basic ANN architectures. In Section III, we provide a brief overview of some of the existing approaches to ML-based propagation modeling, focusing on ray tracing. In Section IV, we present the proposed data-driven indoor propagation modeling framework. Next, in Section V, we provide numerical results comparing our model with synthetic data computed through a ray tracing simulator. We study scenarios where the proposed framework is trained over multiple geometries and frequency bands, and we also explore its applicability to unknown indoor environments and frequencies. Then, in Section VI, we outline how transfer learning can be exploited to calibrate our model, and we compare the predictions of our and other propagation models with measured data. Finally, Section VII concludes this article outlining its main contributions.

II. Theoretical Background

ANNs seek to imitate human intelligence by allowing simple learning components to perform basic computational operations and connect to other learning components. The learning components are referred to as nodes, artificial neurons, or just neurons. In what follows, we briefly discuss the functionality of MLPs and CNNs, focusing on the latter since in this work we are using a convolutional encoder–decoder.

A. Multilayer Perceptrons

In MLPs, the neurons are arranged into layers, where each neuron has input connections originating from the previous layer and output connections pointing toward the next layer. A typical MLP consists of an input layer, a number of hidden layers, and an output layer, as shown in Fig. 1, where $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_m$ are the input features and the outputs of MLP, respectively. With the term input features, we refer to the characteristic physical quantities that affect a quantity of interest (QoI), which is the output of MLP. For instance, the commonly used input features of ML propagation models are the operating frequency, the distance, or the transmitter height, while typical QoIs are the signal level or the PL. The hidden layers consist of neurons that apply nonlinear transformations to their input data. Specifically, the output, $u_j^{(l)}$, of the jth neuron in the lth layer is computed by applying a nonlinear function, g, to the weighted sum of the previous layer neuron outputs, plus a bias term, $b_j^{(l)}$

$$u_j^{(l)} = g\left(\sum_i^{n_{l-1}} \theta_{i,j} u_i^{(l-1)} + b_j^{(l)}\right) \tag{1}$$

where $u_i^{(l-1)}$ is the output of the ith neuron of the previous layer, $\theta_{i,j}$ is the weight associated with $u_i^{(l-1)}$, and n_{l-1} is the number of neurons in the previous layer. Some commonly used nonlinear functions, g, are the rectified linear unit (ReLU), the tanh, and the sigmoid function [28]. The output of MLP is estimated in a similar manner, and the QoI can assume discrete categorical or continuous values, for classification and regression problems, respectively.

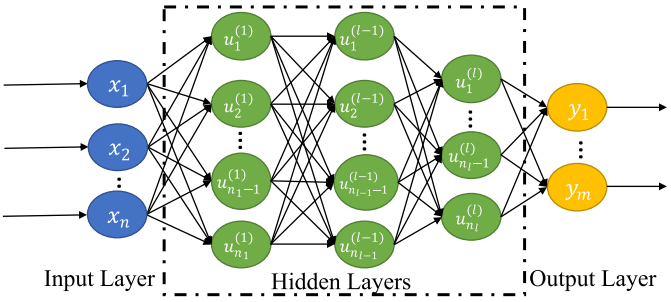

Fig. 1. MLP.

B. Convolutional Neural Networks

CNNs have been widely used in computer vision and in image-processing-related applications. In CNNs, the input data are represented as tensors, and they have three dimensions: the width, W, the height, H, and the number of channels, C. A CNN typically comprises three different layer types: convolutional, pooling, and MLPs. An example of a CNN is shown in Fig. 2, where an RGB image, representing the geometry of an indoor environment, is fed to a convolutional layer.

In a convolutional layer, the input tensor is convolved with n_f different filters. Each filter identifies different hidden features and it consists of stacked 2-D kernels, i.e., 2-D square $f_w \times f_h$ weight matrices. The number of stacked kernels is equal to the number of input channels C, and the number of output channels is equal to the number, n_f, of the individual filters convolved with the input tensor. The result of these convolutions, i.e., the layer output, is known as the feature map. To estimate the elements of feature map, each kernel is slid with a stride, s, over the respective channel of the input tensor, and the dot product between the filter kernel weights and the points of the corresponding overlapping channel subarea is computed. Consequently, the elements of the output feature map are computed as the cross-channel summation of the results of the aforementioned dot products. With respect to Fig. 2, this corresponds to computing the dot product between the orange weight boxes and the pixel intensity values of the overlapping subarea, and then adding the dot products estimated for each one of the three RGB channels. Let X be the input tensor and θ the weights of the o-th filter, then the elements z of the feature map are computed via [28]

$$z_{m,n,o} = \sum_k \sum_i \sum_j \theta_{i,j,k} \cdot X_{m\times s-i, n\times s-j, k} \tag{2}$$

where the summation over i and j captures spatial correlations between the elements of the same channel, while the summation over k enables unveiling correlations between the different channels of the input tensor (cross-channel correlations). The convolutional layers of a CNN are typically followed by pooling layers used to simplify the representation of the encoded feature maps through downsampling and to reduce the number of model's trainable parameters. The most common pooling operation is max-pooling. In a max-pooling layer, a single $f_w \times f_h \times f_c$ filter, where f_c is the number of input

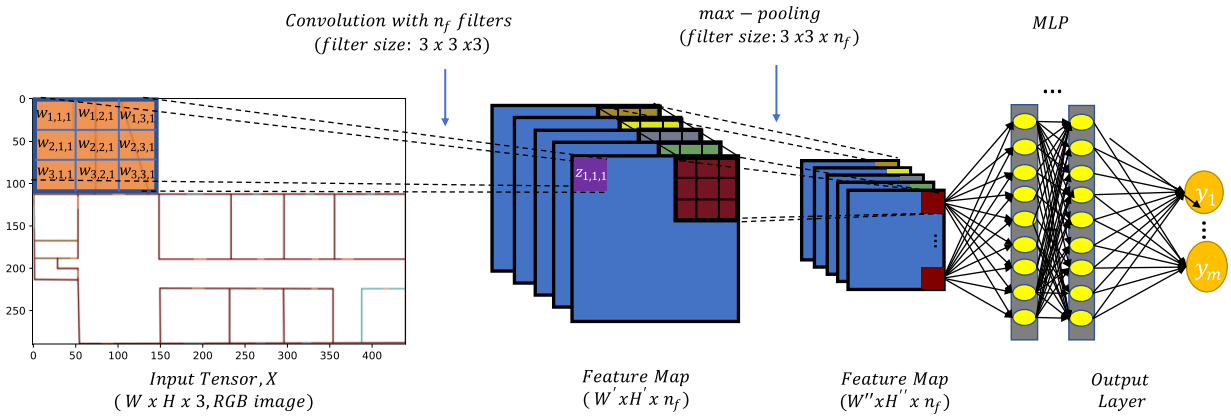

Fig. 2. Typical operations in a CNN. The orange box depicts the kernel of a convolutional filter slid over an RGB image. The different colored squares represent the kernels of a max-pooling filter used to extract the maximum point from a subarea of each channel.

tensor channels, is slid over the received input tensor, and the elements of the output feature map are estimated as the maximum element of the overlapping subarea of each channel. For instance, the different colored squares in Fig. 2 correspond to the kernels of the filter used to extract the maximum point from a subarea of each channel. Finally, CNNs are commonly terminated with MLPs, which are used to estimate the final network output.

The main advantages of CNNs are the weight sharing and the connection sparsity properties. The first means that the same filter weights are applied to different parts of the input tensor, i.e., the same orange box is slid over the entire indoor geometry in Fig. 2. Thus, each convolutional filter can identify certain kinds of hidden features, and more importantly it is not required to compute different weights for every point of the input tensor. The second property signifies that unlike MLPs, where each output neuron receives information from all the neurons of the previous layer, in CNNs, each neuron considers information originating only from a small subarea of the previous layer. Thus, the numerical operations required to compute the output value of a neuron decrease substantially. For instance, for the estimation of $z_{1,1,1}$ in Fig. 2 (purple box), only the top-left side of the indoor geometry is considered.

The connection sparsity property also gives rise to the concept of the receptive field, which depicts the information region of the input tensor that affects the output response of a neuron. Evidently, since each convolutional layer encodes and compresses information of the previous layer, the neurons of deeper layers have an indirect access to a larger area of the input tensor, and thereby a larger receptive field [28]. Hence, the shallower layers within a CNN can detect low-level features of the input tensor, whereas the deeper layers, due to their augmented receptive field, can identify more complicated abstract high-level features.

III. Related Work

The idea of using ANNs to improve the performance of radio propagation modeling tools is not new. Perrault *et al.* [21] used an MLP to calibrate a ray tracer with measurement taken from three different cities taken at 900 and 1800 MHz. The MLP received as input the received signal strength (RSS) provided by the ray tracer and several simulation parameters (transmitter height, number of reflections, and land type). Then, the authors tried to fit the simulated RSS to the measured RSS values and calibrate the ray tracer. More recently, due to their computational efficiency, the use of CNNs for radio propagation modeling has become popular [20], [22]. In [22] and [23], a CNN was trained with images that depicted the buildings of a city, assuming different pixel intensity values according to the building height. Consequently, the PL for different urban environments was simulated through a ray tracing simulator. The city images along with the respective simulated PL values were used as the input and the expected output of the data-driven model, respectively. In [20] and [24], a U-Net-like convolutional encoder–decoder was used. The authors incorporated multiple features to their model by encoding additional information through different input channels. Although these models provided accurate results and they overcame the scalability limitation posed by MLPs, they focus on outdoor propagation modeling and they are not applicable to indoor environments.

Most of the existing AI-based indoor propagation models make use of MLPs [19], [32]. The motivation in the work done in [19] was to reduce the computational cost of ray tracing through a coarse-to-dense grid MLP-assisted scheme. A ray tracing simulation was conducted in an indoor environment assuming a coarse grid discretization. Then, the ray tracing results were used to train an MLP to recognize radio channel characteristics and infer the RSS for other points and coverage "holes," assuming a dense grid discretization. In [32], a similar approach was implemented using real measurements instead of ray tracing data. A CNN-based formulation was presented in [30], consisting of two convolutional layers and an MLP with four hidden layers, aiming to evaluate the characteristics of a millimeter-wave channel. The input data comprised the coordinates of the receiver and the transmitter, while the output vector included various channel characteristics (PL, delay spread, angle of arrival, etc.). The main drawback all the previous approaches share is that they attempt to associate the geometrical coordinates of an indoor environment to telecommunication-related features. As it was shown in [29], this constitutes a fundamental bottleneck in the applicability of such models in geometries other than those in which they are trained.

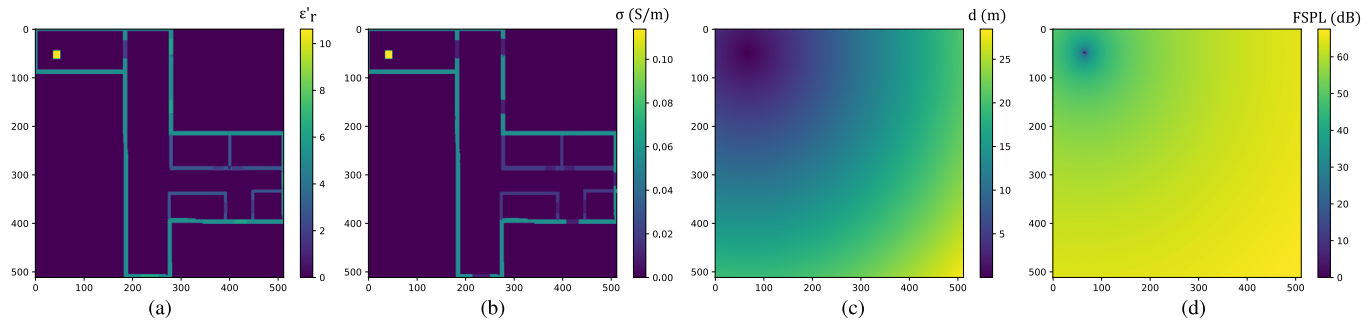

Fig. 3. Example of the input tensor channels for a random sample at an operating frequency of 2 GHz. (a) Permittivity channel. (b) Conductivity channel. (c) Distance channel. (d) FSPL channel.

IV. Problem Formulation and Proposed Model

In this section, we present the details of our data-driven indoor propagation model, called EM DeepRay. We describe the format of the input and output data, the general framework pipeline, and the convolutional encoder–decoder used.

A. Input Features and Target Output

The input features used in our work are: 1) the permittivity; 2) the conductivity of the building materials; 3) the distance between the transmitter and every point within the simulated grid; and 4) the free space PL (FSPL) at every point of the indoor geometry, assuming the absence of the building. The predicted parameter is the PL at every spatial point within an indoor geometry, given a specific transmitter position and building layout. The reasoning behind the input feature selection and the generation of the input tensor is explained in the following paragraphs.

Wave propagation is affected by four basic mechanisms: reflection, transmission, diffraction, and diffuse scattering [16], [33]. It is well-known that the impact of these mechanisms depends on the wavelength λ of the propagating wave. In particular, reflections and transmissions require that the objects found within the propagation environment are electrically large, i.e., their dimensions are much larger than λ. When λ is much larger than the object (sharp edges, small openings), the propagating waves will bend around the object and diffraction will occur. Finally, diffuse scattering requires that abrupt variations in the surface height are an order larger than the wavelength. The impact of these mechanisms also depends on the material proprieties, which in turn are related to the radio signal frequency [34].

The electromagnetic properties of a material can be quantified by its permittivity and its permeability. In this work, we assume that all the construction materials found within the simulated indoor geometries are nonmagnetic, i.e., they have a constant permeability equal to μ_0. The permittivity is formulated as: $\epsilon = \epsilon_r \epsilon_0 = (\epsilon_r' - j^{\sigma}/_{\epsilon_0 \omega})\epsilon_0$, where ϵ_r and σ are the relative permittivity and the conductivity, respectively. Thus, to consider how different materials affect wave propagation, we use two input channels that convey information regarding the relative permittivity. Specifically, the first channel depicts the real part of ϵ_r at every point of the simulated grid, and the second includes the value of the conductivity. An example of these channels is shown in Fig. 3(a) and (b), respectively. The conductivity is modeled as $\sigma = cf^d$, where f is the frequency of the propagating wave in GHz. The values of ϵ_r', c and d, are derived from the ITU-R P.2040-1 Recommendation [34], and they are shown in Table I. The first channel allows our model to understand the presence of an object and infer the strength of the reflected and the transmitted components of an electromagnetic wave impinging onto it. The second channel accounts for the attenuation that an electromagnetic wave undergoes while it propagates through an absorbing medium. Also, to indicate the transmitter position within the simulated grid, the values of ϵ_r' and σ around the transmitter's position are set equal to twice the maximum values of ϵ_r' and σ.

TABLE I
Material Parameters

Material	ϵ_r'	c	d
Concrete	5.31	0.0326	0.8095
Plasterboard	2.94	0.0116	0.7076
Wood	1.99	0.0047	1.0718
Glass	6.27	0.0043	1.1925
Brick	3.75	0.038	0
Air	1	0	0

Another important parameter that affects wave propagation in indoor environments is wall thickness. In outdoor ray tracing simulations, building walls are modeled as semi-infinite spaces due to the high losses, and thus the penetration into buildings or the multiple reflections within the building facets can be omitted. In indoor environments, this assumption does not hold, since the walls are thinner than the building facets, and for materials such as plasterboard or wood, the losses are considerably smaller. Hence, objects such as walls, doors, and windows are modeled as slabs, and consequently the reflection and transmission coefficients depend on slabs' thickness. To account for that, we use a third channel which depicts the physical distance between the transmitter and every point in the simulated grid. An example of the distance channel is shown in Fig. 3(c). Thus, when a convolutional kernel is applied to a subarea of the input tensor, apart from detecting the wall type through the first two channels, it can also consider the wall length via the difference between the values of the third channel.

Furthermore, the third channel can also provide a good estimate about the deterioration of the signal over distance.

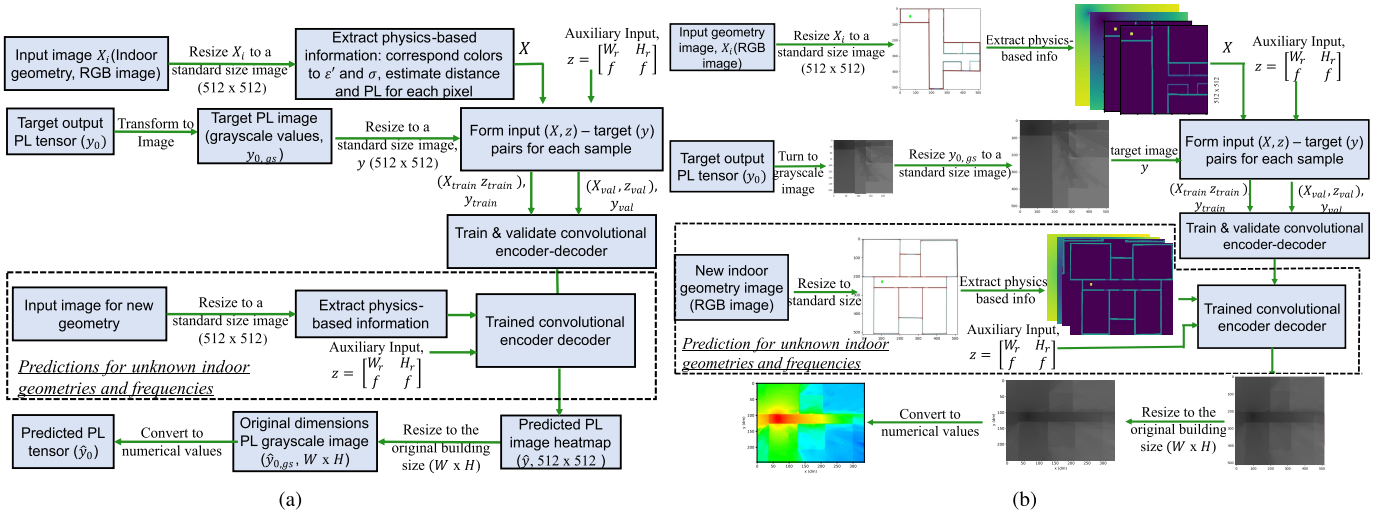

Fig. 4. EM DeepRay architecture. (a) Block diagram of EM DeepRay architecture. (b) Schematic of EM DeepRay functionality.

To further enhance the awareness of our model regarding the impact of distance on wave propagation, we add a fourth channel which includes the FSPL for each point in the simulated grid. We note that if our model was trained only over one frequency, the fourth channel would have been a simple transformation of the third channel, and thus this information would have been redundant. However, EM DeepRay is trained over multiple frequencies, and hence the fourth channel helps our data-driven model to unveil correlations over the frequency and space domains.

The target output of EM DeepRay is a tensor including the PL at every point of the simulated grid. We note that a similar treatment can be applied for another QoI such as the RSS or the phase delay. As shown in [29], each channel of the output tensor corresponds to different sampling heights. In this work, we use a single output channel, representing the PL at the horizontal plane at a height equal to 1.5 m. The target PL values are computed using a ray tracing simulator [35]. The goal is to train a convolutional encoder–decoder which encodes the input tensor and then decodes it as a PL heatmap, i.e., to find a mapping between the material properties, physical distance, and frequency space and the PL space.

B. Workflow

The block diagram of the EM DeepRay, shown in Fig. 4(a), preserves the same idea as the initial DeepRay framework [29]. In the initial framework, the simulated results from a ray tracing simulator are leveraged to train a convolutional encoder–decoder to learn how to transform a blueprint of the indoor geometry to a PL heatmap. For each ray tracing simulation, we obtain an image, X_i, representing the geometry of an indoor environment and a tensor, y_o, comprising the simulated PL values at each point of the indoor geometry. The target PL tensor is converted into a grayscale image, $y_{o,gs}$, and consequently it is resized to a standard size image, y. Similarly, the input geometry image is also resized to a standard size image X_c. Although a convolutional encoder–decoder can be trained with varying sized images, using standard size images facilitates the training procedure [36]. To avoid a substantial stretching or squeezing when resizing is applied to the building layouts and the target PL image, the standard size selected in this work is 512 × 512. The resized input geometry and PL images are provided as an input–target pair to a convolutional encoder–decoder that performs a pixel-to-pixel prediction, translating the blueprint of an input geometry to a PL heatmap.

The main difference between the two frameworks is the form of the input tensor. The initial DeepRay framework uses a blueprint of the indoor geometry, and the convolutional encoder–decoder learns to identify how different construction materials affect wave propagation, by representing each material with a distinct color. In this work, we use the blueprint of the input geometry to extract physics-based information that will augment the performance of the convolutional encoder–decoder and allow it to make more accurate predictions. Specifically, as discussed in Section IV-A, for each indoor geometry image, X_i, we derive an input tensor, X, with four channels representing: 1) ϵ_r' and 2) σ at each point of the grid; 3) the physical distance between the transmitter and each point; and 4) the FSPL for each point. Hence, first we resize the initial input geometry image to a standard size image, and then we use the resized image to compute the values of the four input channels of Fig. 3. The first two channels are derived by matching the colors of the blueprint, i.e., different RGB pixel values, to the values of Table I. The physical distance, $d_{i,j}$, for the (i, j)-th pixel is computed as

$$d_{i,j} = R\sqrt{W_r(x_{Tx} - i)^2 + H_r(y_{Tx} - j)^2} \tag{3}$$

where R is the spatial resolution of the ray tracing simulation, (x_{Tx}, y_{Tx}) corresponds to the transmitter position in the simulated grid, and $W_r, H_r = W, H/512$ are the width and height resizing ratios, respectively. Consequently, the values of the fourth channel are estimated as

$$FSPL_{i,j} = 20\ log_{10}\left(\frac{4\pi d_{i,j} f}{c_o}\right) \tag{4}$$

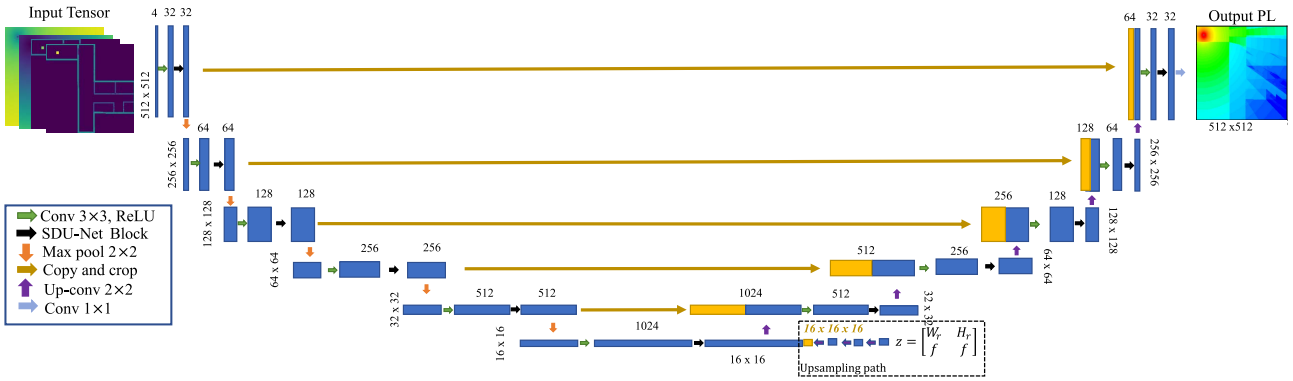

Fig. 5. Convolutional encoder–decoder used in EM DeepRay to encode a physics-based input tensor and decode it as a PL heatmap.

where f is the operating frequency in Hz and c_o is the speed of light.

A schematic representation of this process is shown in Fig. 4(b). As can be observed, the initial input image is resized to a 512 × 512 image, and it is used to extract the four-channel tensor X, which is input to the convolutional encoder–decoder. Apart from the input tensor X, the convolutional encoder–decoder receives an auxiliary input z, which conveys information about the frequency and the resizing ratios, W_r and H_r, of each sample. The details of the incorporation of the auxiliary input with the input tensor are outlined in Section IV-C. The target PL tensor is turned to a grayscale image, assuming values between 0 and 255, and the grayscale image is resized to have the same size as X (i.e., 512 × 512). The resized geometry image along with the auxiliary input tensor form a pair of input values, $(X,\ z)$, while y is the target value. Each input–output pair accounts for a sample used to train the convolutional encoder–decoder.

Once trained, the convolutional encoder–decoder can be used to directly predict the PL for new geometries or frequency bands that are not included in the training dataset (dashed box in Fig. 4(a) and (b)). Predictions for new geometries require only an image of the indoor environment, which is used to extract the physics-based input tensor, as shown in Fig. 4(b). The request of an image depicting the input geometry is not undue, since most of the existing commercial ray tracing software use such a representation [17], [35]. In addition, our approach can be further facilitated by recent research which allows computer-aided design (CAD) floor plans to be turned into blueprints [37]. We also note that the extraction of the physics-based input tensor does not entail any complex operations, only if statements and matrix multiplications whose computation requires a few milliseconds.

Given the physics-based input tensor, the convolutional encoder–decoder predicts a PL heatmap image, $\hat{y}$, for the desired input geometry. The predictions of the convolutional encoder–decoder are pixelwise, and the size of the output tensor is 512 × 512. Thus, after prediction, we use a bilinear interpolation to resize the output image back to the initial dimensions W and H of each geometry. As a final step, the grayscale PL images are converted into numerical values. The latter simply implies perceiving the elements of the resized input image as dB instead of pixel intensity values.

C. Convolutional Encoder–Decoder

Conventionally, CNNs are terminated by MLPs, as shown in Fig. 2. There are three main drawbacks associated with this approach: 1) MLPs require a fixed-size input; 2) they neglect the spatial and cross-channel correlations; and 3) they are computationally expensive as they have more parameters [36]. This motivated the development of fully convolutional networks (FCNs), which are able to perform pixel-to-pixel predictions [36]. In FCN architectures, the MLP at the end of the network is replaced by a series of upsampling and convolutional layers. The absence of the MLP translates to a smaller number of trainable parameters and a higher efficiency.

A convolutional encoder–decoder is an FCN that consists of two basic components: 1) an encoder, used to downsample the input tensor and compress its context, and 2) a decoder which is used to recapture the input tensor details and reconstruct it in the form of the target output tensor. Convolutional encoders–decoders may also include skip connections between the encoding and the decoding path, as shown in Fig. 5 (dark yellow arrows), to retain information that might be lost during the encoding procedure. In this work, we use the convolutional encoder–decoder depicted in Fig. 5, which is a slightly modified version of the SDU-Net architecture introduced in [31] (we include an additional upsampling path).

The SDU-Net used has five encoding and decoding layers, i.e., five downsampling and upsampling operations are applied, denoted by orange and purple arrows at the encoding and the decoding branches, respectively. As can be seen in Fig. 5, the left path of SDU-Net is used to encode and downsample the input tensor containing information about the indoor geometry, while the right path upsamples the encoded tensor and transforms it to a PL heatmap. At the end of the encoding path, we add an upsampling path used to upsample the auxiliary input z. Consecutive upsampling operations are applied to z to increase its size and concatenate it with the encoded representation of the input tensor. This will allow the SDU-Net to understand the operating frequency and also be aware of the

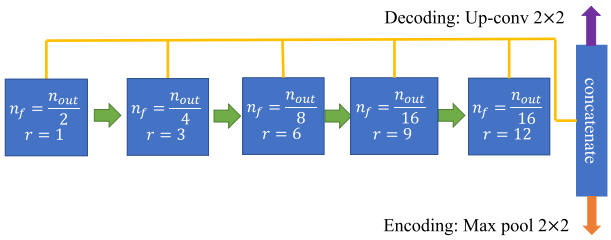

Fig. 6. Structure of the SDU-Net block used at each encoding/decoding layer [31].

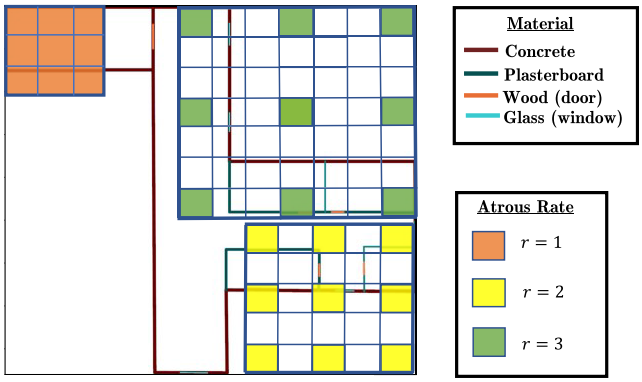

Fig. 7. Example of convolutional filters with different atrous rate r applied to the image of an indoor environment.

resizing ratio for each sample. This information is not included in the input tensor for computational efficiency, i.e., there is no need to reserve an entire channel for scalar variables. Instead, it is considered during the decoding by fusing the upsampled z with the encoded input tensor just before the decoding procedure begins.

At the encoding path, each layer includes a 3×3 standard convolution, followed by the SDU-Net block, shown in Fig. 6, and a 2×2 max pooling with stride equal to 2. At the decoding branch, the feature map of the previous layer is first upsampled, and then it is concatenated with the feature map of the respective layer of the encoding branch (yellow boxes shown at each layer of the decoding branch). Consequently, a 3×3 standard convolution is applied to the concatenated tensor, and the resulting feature map is forwarded to the SDU-Net block and moved upward to the next layer. All the convolutional layers use an ReLU activation function, except for the last 1×1 convolution which uses a linear activation function. The number of convolutional filters is increased or decreased by a factor of 2 at each encoding or decoding layer, respectively. At the top layer of the decoding branch, a 1×1 convolution is used after the SDU-Net block to derive the PL heatmap image. The number of filters for the final 1×1 convolution is equal to the number of the desired output channels, i.e., one.

The SDU-Net block, as shown in Fig. 6, comprises five cascaded atrous convolutions, each one using a different number of filters and a different atrous rate. The feature maps of different atrous convolutions are concatenated and forwarded to the next layer after being down- or upsampled. Atrous convolution, also known as dilated convolution, is a generalized case of the standard convolution operation. Its popularity has recently increased, since it can be used to efficiently augment the receptive field of each neuron without increasing the number of model parameters [38]. In atrous convolution, the kernel is widened and its sparsity is increased by introducing blank spaces between the kernel elements. The distance between neighboring nonblank elements of the kernel is $r - 1$, where r is the atrous rate. The parameters for each atrous convolution are shown in Fig. 6, where r is the atrous rate, and n_{out} denotes the number of filters at the next layer (which is twice or half the number of filters at the current encoding or decoding layer, respectively).

An example of 3×3 atrous convolutions with different values of r applied to the image of an indoor environment is shown in Fig. 7. It can be observed that as r increases, the receptive field is augmented, since the convolutional kernel has an enhanced field of view over the indoor geometry. Indeed, the area covered by the convolutional kernel with $r = 1$ (standard, nondilated, convolution, orange box) is substantially smaller than that of the kernel with $r = 3$. That comes at no increase in the overall computational cost, since the number of trainable parameters remains the same. Furthermore, the use of various atrous convolutions with different atrous rates enables the aggregation of information originating from different spatial scales and the identification of correlations between distant points. Moreover, due to multiple layers of the SDU-Net architecture, it is also possible to capture multiscale correlations at different resolutions (since the size of the

$$MAE = \frac{1}{NWH} \sum_{n=1}^{N} \sum_{i=1}^{W} \sum_{j=1}^{H} |y_{o,(n)}(i,j) - \hat{y}_{o,(n)}(i,j)| \tag{5}$$

$$MAPE = \frac{1}{NWH} \sum_{n=1}^{N} \sum_{i=1}^{W} \sum_{j=1}^{H} \frac{|y_{o,(n)}(i,j) - \hat{y}_{o,(n)}(i,j)|}{|y_{o,(n)}(i,j)|} \tag{6}$$

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N} \sum_{i=1}^{W} \sum_{j=1}^{H} \left(y_{o,(n)}(i,j) - \hat{y}_{o,(n)}(i,j)\right)^2}{NWH}} \tag{7}$$

$$r_p = \frac{\sum_{n,i,j}^{N,W,H} \left(y_{o,(n)}(i,j) - \overline{y}_{o,(n)}\right)\left(\hat{y}_{o,(n)}(i,j) - \overline{\hat{y}}_{o,(n)}\right)}{\sqrt{\sum_{n,i,j}^{N,W,H} \left(y_{o,(n)}(i,j) - \overline{y}_{o,(n)}\right)^2 \sum_{n,i,j}^{N,W,H} \left(\hat{y}_{o,(n)}(i,j) - \overline{\hat{y}}_{o,(n)}\right)^2}} \tag{8}$$

input tensor is decreased by a factor of 2 at each encoding layer). The augmented receptive field and the potential of capturing multiscale correlations (using various r values) at multiple resolutions (due to the multiple encoding and decoding layers) render the SDU-Net an exemplary convolutional encoder–decoder for radio propagation modeling tasks.

D. Evaluation Metrics

To evaluate the performance of DeepRay, we use four error metrics: the mean absolute error (MAE), the mean absolute percentage error (MAPE), the root mean square error (RMSE), and the Pearson correlation coefficient, r_p, defined as follows: (5)–(8), as shown at the bottom of the previous page, where $y_{o,(n)}(i, j)$ and $\hat{y}_{o,(n)}(i, j)$ are the ray tracing and the predicted PL values for the nth sample at the spatial point (i, j), respectively, $\overline{y}_{o,(n)}$ and $\overline{\hat{y}}_{o,(n)}$ are the corresponding mean values for that sample, and N represents the number of samples. We note that the error is measured at the end of the proposed framework, i.e., it is not pixelwise but it is estimated with respect to the actual PL values.

V. Comparison With Simulated Data

In this section, we proceed to the implementation of EM DeepRay, comparing its PL predictions with simulated data. To create the synthetic data to train EM DeepRay, we use the Ranplan Professional ray tracing simulator, developed by Ranplan Wireless [35]. Ranplan Professional is a robust radio propagation engine, supporting radio propagation simulations for both indoor and outdoor environments. It enables easy and efficient wireless network design of entire floors or even buildings and it supports a large number of different communication technologies (WiFi, 5G New Radio, IoT). The software has been widely used for actual network planning, which provides us an assurance regarding its reliability.

To train and validate EM DeepRay, we conduct ray tracing simulations at 20 different indoor environments. The buildings considered include simple and more complicated indoor environments, with more than 20 subrooms in the same building floor, and the walls are made of various construction materials (concrete, brick, plasterboard, wood, and glass) and have various thicknesses. For each building, we consider transmitting devices operating at three different frequencies: 1) 0.433; 2) 2; and 3) 3.7 GHz, which correspond to frequency bands used in IoT and 5G systems. To create our dataset, we place a sole transmitter operating at a given frequency within a building, and then conduct a ray tracing simulation only for this transmitter. Once the ray tracing simulation is finished, we change the transmitter position and/or the frequency, and we run a new ray tracing simulation to create a new sample. This corresponds to consecutively sampling the PL response in an environment of interest for a given transmitter position and frequency.

For all the simulations, the transmitter antenna is 1.5 m above the floor and has an omnidirectional beam pattern. The resolution of the ray tracing simulations is set to 0.1, i.e., we sample the PL value once per 0.1 m. For each frequency band, we run approximately 35 different ray tracing simulations at each building, i.e., 35×3 samples per building, assuming a different transmitting device position within the indoor geometry for each sample. We use 80% of the samples to train DeepRay and 20% to validate its performance, which is a commonly used data splitting ratio in ML problems [28]. Once the data are split into two sets, the training dataset is augmented by flipping the indoor geometry, along with the transmitter's position within it, and the PL heatmap images left, right, and downward. The initial and the flipped input images exhibit a symmetry, and hence one should expect that the ray-tracing results should be the same. Indeed, the intersections between the rays and the walls, the respective reflection and transmission coefficients, and the diffracting edges remain the same, and thus the flipped PL images are equivalent to the ray tracing simulation results for the flipped geometry. This allows an effective increase in the training set size by a factor of 3 without conducting any extra simulations. We note that EM DeepRay is validated with samples from known geometries and frequency bands, i.e., from indoor environments and frequency bands included in the training dataset, but for unknown transmitting device positions within these environments.

To test EM DeepRay, we consider three cases, aiming to explore how well our model: 1) can generalize to new geometries, not included in the training dataset; 2) can infer the PL for frequency bands other than those of the training set; and 3) behaves in a combination of 1) and 2). We refer to these cases as Tests 1, 2, and 3, respectively. The case studies are summarized in Table II, where with the term "known" and "unknown" we indicate either included or not included in the training dataset. To implement Test 1, we consider five new indoor environments, generating about 35 samples at each building, for each frequency band of the training dataset (0.433, 2, and 3.7 GHz). We underline that EM DeepRay has no prior information regarding PL distribution within these indoor environments, and it will infer the PL heatmaps based on the weights estimated during the training phase. For Test 2, we assume transmitting devices operating at 866 MHz and 2.6 GHz, and we randomly place approximately 15 devices for each frequency at each one of the 20 buildings used to train and the validate our model. Finally, for Test 3, we consider transmitting devices operating at 866 MHz and 2.6 GHz at the five buildings used in Test 1, i.e., for buildings and a frequency band not included in the training dataset, taking approximately 40 samples per building and frequency.

Our data-driven model is trained on a Nvidia Quadro RTX 8000 GPU over Tensorflow, using the Adam optimization algorithm for 250 epochs, with the learning rate set to 0.0005, and a batch size of 4. Before passing the input data to our model, a min–max normalization is applied separately at every input channel [28]. The loss function to be optimized is RMSE. We also explored the use of MAE, but the performance of our model was slightly worse, and thus we present results only using RMSE as a loss function. During training, the loss function is minimized with respect to y and $\hat{y}$, i.e., based on the pixel intensity values of ray tracing and the predicted PL images. However, the error metrics are estimated with respect to ray tracing and the predicted actual PL values (i.e., between

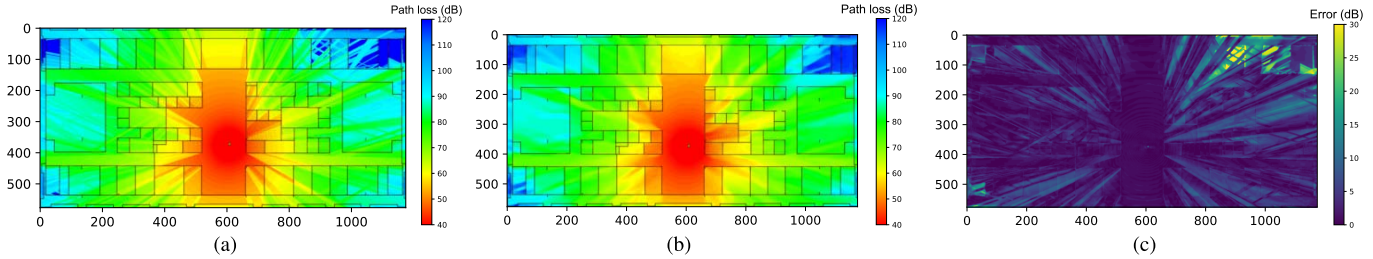

Fig. 8. Comparison between the PL heatmap simulated through (a) ray tracing and predicted by (b) EM DeepRay for a validation set sample at 433 MHz (known building and frequency, but unknown transmitter position). (c) Error map depicts the absolute error between (a) and (b).

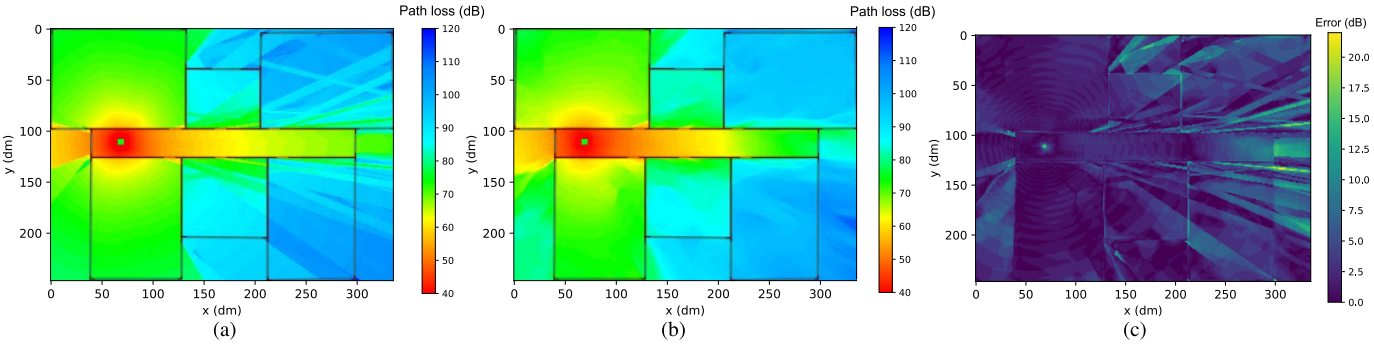

Fig. 9. Comparison between the PL heatmap simulated through (a) ray tracing and predicted by (b) EM DeepRay for a Test set 1 sample at 2 GHz (known frequency band, but unknown indoor geometry). (c) Error map depicts the absolute error between (a) and (b).

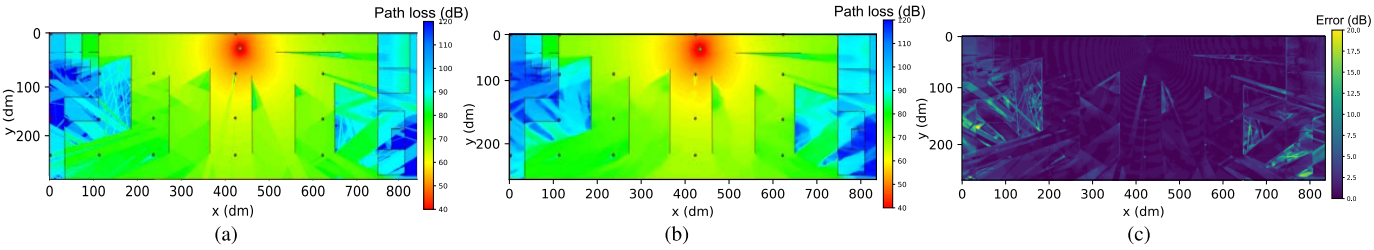

Fig. 10. Comparison between the PL heatmap simulated through (a) ray tracing and predicted by (b) EM DeepRay for a Test set 2 sample at 2.6 GHz (known indoor geometry, but unknown frequency band). (c) Error map depicts the absolute error between (a) and (b).

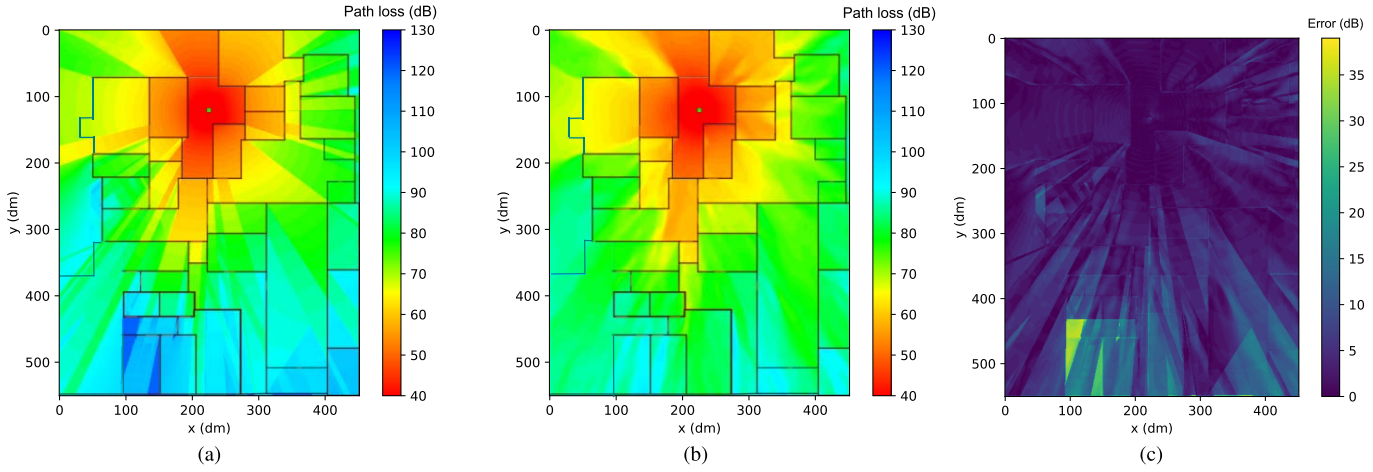

Fig. 11. Comparison between the PL heatmap simulated through (a) ray tracing and predicted by (b) EM DeepRay for a Test set 3 sample at 868 MHz (unknown indoor geometry and frequency band). (c) Error map depicts the absolute error between (a) and (b).

y_o and $\hat{y}_o$, in dB). The training takes almost 2 h, while a prediction for a single sample requires about 100 ms, including the extraction of the physics-based input tensor.

The overall error metrics between the simulated and predicted PL values for all the case studies are presented in Table III. The error metrics are negligible for the training set, with the RMSE and the MAE being equal to 1.2 and 0.99 dB, respectively. This implies that the proposed framework can indeed learn to translate a map of physics-based information to a PL heatmap. The validation set error metrics

TABLE II
CASE STUDIES: KNOWN AND UNKNOWN INDICATE INCLUDED OR NOT INCLUDED IN THE TRAINING DATASET

	Transmitter Position	Geometry	Frequency
Training	Known	Known	Known
Validation	Unknown	Known	Known
Test 1	Unknown	Unknown	Known
Test 2	Unknown	Known	Unknown
Test 3	Unknown	Unknown	Unknown

TABLE III
EM DEEPRAY PERFORMANCE FOR EACH CASE STUDY

	RMSE (dB)	MAE (dB)	MAPE	r_p	Samples
Training	1.2	0.99	0.01	0.97	1680
Validation	4.21	3.26	0.03	0.937	420
Test 1	5.05	3.86	0.04	0.94	525
Test 2	4.77	3.63	0.04	0.95	600
Test 3	5.85	4.63	0.05	0.94	400

are approximately 3 dB larger than these of the training set, but they still assume low values. More importantly, the good resemblance between the simulated and predicted data is also preserved for the three different test cases, as is indicated by the values of r_p that are very close to 1. Also, Tests 1 and 2 yield approximately an RMSE of 5 dB and an MAE of 3.8 dB, while for Test 3 the respective values are 5.6 and 4.5 dB. Furthermore, the MAPE assumes very small values for every scenario. Thus, in addition to learning to infer the signal attenuation within a given geometry and frequency band, our model is generalizable, and it can provide accurate estimates of the PL for new indoor geometries and frequencies. By exploring the generalizability of our model to buildings and frequencies separately, it is easier to understand its potential limitations, and to where these could be attributed. Also, in Test 3, by studying the same buildings and frequency bands as in Tests 1 and 2, we can be confident that any error is not owning to the use of new unknown buildings or frequencies, but it can be attributed to the simultaneous generalization over new buildings and frequencies.

In Figs. 8–11, we visualize the results for a random sample from the validation set and from test sets 1–3, respectively. Figs. 8(a)–11(a) show the simulated ray tracing "ground-truth" for each sample, while the corresponding PL heatmap predicted by EM DeepRay is illustrated in Figs. 8(b)–11(b). We note that DeepRay outputs solely the PL heatmap, and to illustrate the input geometry we consequently impose its blueprint on top of the PL heatmap. As can be observed, the similarity between the simulated and predicted PL heatmaps is very high for all the cases. The absolute error maps in Figs. 8(c)–11(c) depict the absolute error, $|y_o(\mathrm{i,j}) - \hat{y}_o(\mathrm{i,j})|$, at each spatial point (i, j) within the simulated indoor environment. A common trend is that the predicted PL exhibits the largest absolute errors at positions far away from the transmitting device. For instance, as can be seen in Fig. 11(c), the largest errors are found on the lower left area of the floor. That is more than 40 and 10 m away from the transmitter, in the horizontal and vertical directions, respectively. This is encouraging, since when a user is located far away from a transmitter it is likely that they are served by another device, and thus these areas do not have significant impact on network design decisions.

VI. COMPARISON WITH MEASURED DATA AND CALIBRATION THROUGH TRANSFER LEARNING

In Section V, we demonstrated that EM DeepRay can be trained with physics-based data of various indoor environments and eventually learn to predict the PL within them. Once trained, EM DeepRay can be used as a standalone propagation model to furnish estimates of the PL for an arbitrarily complex indoor geometry at a given frequency. The credibility of our model highly depends on the data used to develop it. These data are only an approximation of the actual signal attenuation and they are themselves subject to errors. However, the purpose of an ML-based propagation model is to deliver results that closely resemble actual rather than synthetic data.

In this section, we address this issue by outlining a simple, yet efficient, approach that allows the calibration of EM DeepRay through transfer learning. The aim of transfer learning is to leverage knowledge accrued to tackle a certain problem and use it to solve a different problem, which is related back to the initial problem [28]. In our case, the first problem is the design of an ML-based propagation model that can predict the PL in indoor environments, while the second problem is to make the ML-based propagation model realistic. The motivation behind the use of transfer learning, rather than training EM DeepRay simultaneously with synthetic and measured data, is that the measured data are scarce and sparse. Indeed, measurement campaigns are time-consuming and expensive, and usually the results from large-scale campaigns are not publicly available. In addition, typically only a small number of measured RSS values are available for each indoor environment, since these campaigns are realized by moving the receiving device around within the area of interest and recording the RSS at some sparse points.

Hence, if a dataset with both the simulated and measured data is used to train an ML propagation model, it would comprise a limited number of measurement scenarios (compared with the simulated ones) with only a few recordings per scenario. That would render the training challenging, as it would be necessary to find a balance between the importance of the measured over the simulated data. Instead, treating the calibration as a distinct problem enables handling a limited quantity of sparse data in a more efficient manner. In particular, to calibrate our model we use the same framework shown in Fig. 4(a); however, the target output tensor is now different. As shown in Fig. 12, for calibration, the target output tensor comprises the recorded measurements at some points within the simulated grid, instead of the simulated ray tracing PL values. To distinguish the target output tensors of the calibration framework, we use the subscript c, referring to the initial and the resized target output as $y_{c,0}$ and y_c, respectively. Thus, the calibration process constitutes a retraining of the pretrained model obtained in Section V to adapt its PL predictions to match the measured PL values of y_c.

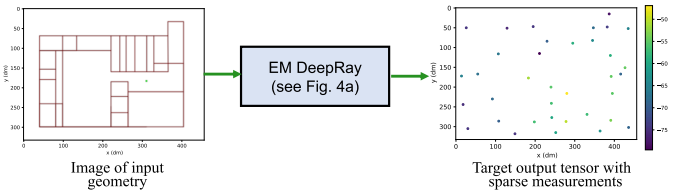

Fig. 12. Calibration framework; the dots in the target tensor correspond to points in the indoor geometry at which the RSS is measured, and their color represents RSS intensity.

A significant difference between the target output tensors of the frameworks, as shown in Figs. 4(a) and 12, is that most elements of $y_{c,0}$ and y_c are zeros, i.e., no measurements are recorded for these points. For that reason, it is not possible to use the same loss function as the one used in Section V. More specifically, the operation of EM DeepRay is equivalent to applying a mapping function f, parameterized through a set of learnable weighs Θ, to the initial input tensor X to derive the output PL image heatmap, i.e., $\hat{y} = f(X|\Theta)$. The values of Θ are computed iteratively, such as to minimize the difference between the pixel intensity of the ray tracing "ground truth" and the predicted PL image heatmap

$$\min_{\Theta} \left(\frac{1}{NWH} \sum_{n,i,j=1}^{N,W,H} \left(y_{(n)}(i,j) - \hat{y}_{(n)}(i,j) \right)^2 \right)^{1/2} \tag{9}$$

where $y_{(n)}(i,j)$ and $\hat{y}_{(n)}(i,j)$, are the ray tracing and the predicted pixel intensity values, respectively, for the nth sample at pixel (i,j). Evidently, if y is zero at most points, for (9) to be minimized, the values of Θ should be selected such as $\hat{y}$ is also zero at these points. Thus, using the same loss function for y_c, which has a few nonzero points, will force the pretrained EM DeepRay to breakdown. To overcome this limitation, during the calibration when (9) is computed for y_c and $\hat{y}_c$, their difference is multiplied with a term Q_c defined as

$$Q_c(i,j) = \begin{cases} 1, & \text{if } y_c(i,j) \neq 0 \\ 0, & \text{if } y_c(i,j) = 0 \end{cases} \tag{10}$$

and hence the only elements that contribute to the loss function are those for which measured data exist. This allows us to slightly modify the preestimated weights Θ during the calibration (retraining) of our model and compute some new weights Θ', in a way that $\hat{y}_c = f(X|\Theta')$ matches only the nonzero values of y_c. We note that the role of Q_c is to consider only the nonzero points of y_c during the computation of Θ'; however, that does not imply that the predictions for other points of the simulation grid will remain intact. That is to say, that Θ' is estimated based only on a few points, but the change in the weights affects the predictions for the entire grid.

To demonstrate the effectiveness of our approach, we generate the indoor geometry of [39] and provide it as an input to DeepRay. The operating frequency is 868 MHz. Note that detailed information about the measurement setup can be found in [39]. In the absence of detailed information regarding the material types, all the walls found within the geometry are assumed to be made of concrete and are 10-cm thick. Again, the input geometry image is resized and it is used

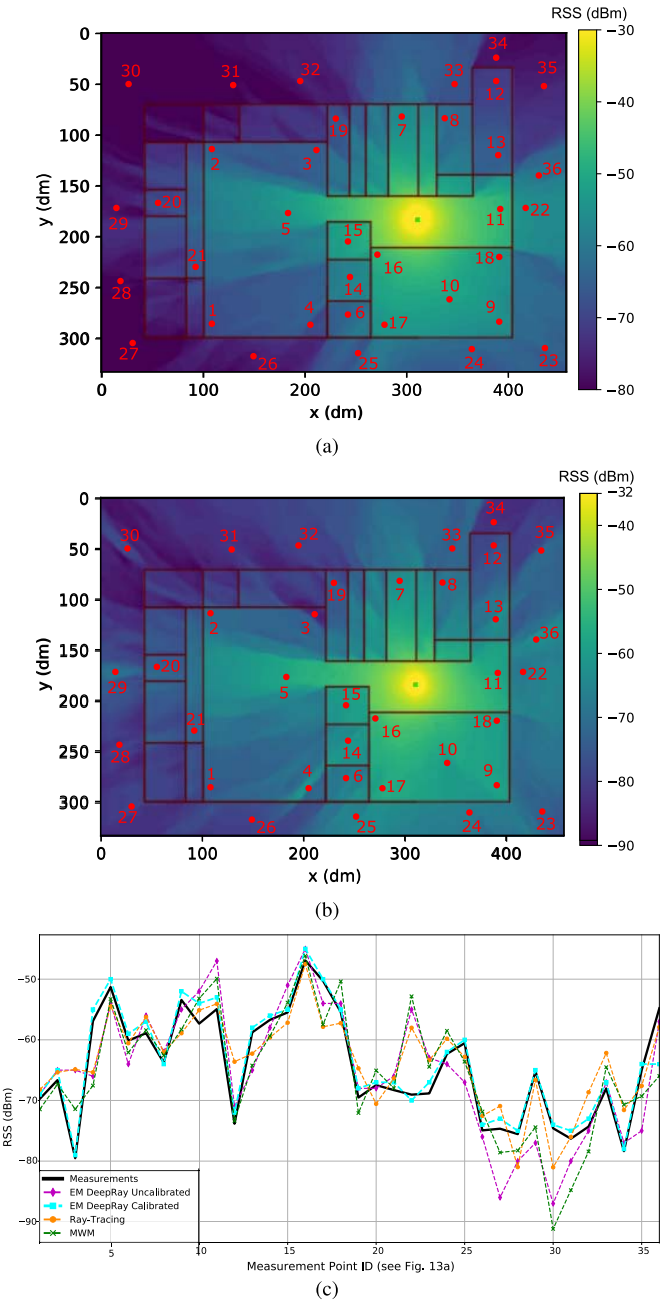

Fig. 13. Predicted RSS, and comparison with measured data [39] (a) DeepRay RSS prediction before calibration for the indoor geometry of [39]. (b) DeepRay RSS prediction after calibration for the indoor geometry of [39]. (c) Comparison between the simulated and measured RSS.

to derive the physics-based input tensor. The target output, $y_{c,o}$, is turned into an image and it is also resized to have the same size as the input tensor. The target output tensor has only 36 nonzero points, i.e., we have measurements for 36 points within the simulated geometry. We note that the measurements usually depict the RSS, while EM DeepRay predicts the PL. To account for that, the elements of $y_{o,c}$ are calculated as $Tx_{power} - RSS_{measured}$, where Tx_{power} is the transmitting power. Then, the two tensors along with the auxiliary vector are passed to the SDU-Net which performs a pixel-to-pixel regression, updating its weights considering only the nonzero points of the target output image tensor, y_c,

TABLE IV

ERROR OF EACH PROPAGATION MODEL COMPARED WITH MEASUREMENTS

	EM DeepRay	RT	MWM	Calibrated EM DeepRay
RMSE (dB)	6.10	5.24	6.21	1.69
MAE (dB)	4.65	4.33	4.78	1.22
Computational time (sec)	0.09	1280	194	15.1

TABLE V

CALIBRATION RESULTS USING A SMALLER NUMBER OF RANDOM TRAINING POINTS, AND ESTIMATING THE ERROR WITH RESPECT TO THE REMAINING MEASUREMENT POINTS

Number of Training Points	4	8	16	27
RMSE (dB)	4.46	3.78	3.32	2.57
MAE (dB)	3.62	3.24	2.45	2.01

as discussed earlier. The SDU-Net is retrained using the Adam optimization algorithm. To avoid abrupt changes in Θ, we train the SDU-Net over fewer epochs and we use a smaller learning rate than that of Section V. Expressly, the learning rate is equal to 10^{-5} and the training lasts for 50 epochs. Finally, to prevent overfitting, we add an L_2 regularization at each convolutional layer, setting the regularization parameter equal to 0.01 [28]. The calibration procedure takes 15 s, since the SDU-Net is retrained over a single sample. Once the SDU-Net is retrained, we obtain the predicted PL tensor $\hat{y}_{c,o}$ and we subtract it from the transmitting power to derive the predicted RSS values.

The predicted signal level by DeepRay is benchmarked against ray tracing and the multiwall model (MWM) [39]. The RSS predicted by DeepRay before calibration is shown in Fig. 13(a), with the blueprint of the input geometry printed on top of it. The predicted RSS after calibration is shown in Fig. 13(b). The red dots in Fig. 13(a) and (b), numbered from 1 to 36, correspond to the points at which the measurements were taken. The measured, simulated, and predicted RSS for each point are shown in Fig. 13(c), where the x-axis signifies the corresponding red dot position shown in Fig. 13(a). A comparison of the error metrics for the three propagation models is presented in Table IV.

Prior to calibration, ray tracing demonstrates the smallest errors yielding an RSME and the MAE being equal to 5.2 and 4.3 dB, respectively. The calculated error values for EM DeepRay and MWM are approximately 1 dB larger. We remark that EM DeepRay is not trained in this geometry and it can directly make predictions by just using a blueprint. Also, the fact that our model can provide estimates of the PL that are in close agreement with measurements can assure us that the number of training samples used in Section V is sufficient to develop a credible data-driven model. A significant advantage of EM DeepRay over both ray tracing and MWM is the substantially computational time required to compute the PL values. Indeed, our model's estimation is based solely on multiple matrix multiplications that can be executed within a few milliseconds in a GPU. On the other hand, ray tracing requires to determine all the rays that can reach a receiving point, while MWM needs to estimate the number of walls between the transmitter and each point in the simulation grid through Bresenham's line algorithm [40]. The substantial difference in the computational time can be critical when it comes to optimal network planning, where multiple simulations are conducted within the same geometry, aiming at meeting certain quality of service requirements.

After calibration, the predicted RSS values for EM DeepRay show a close correspondence with the measured data, exhibiting an RMSE and an MAE equal to 1.69 and 1.22 dB, respectively. The computational time for the calibrated EM DeepRay is shown in Table IV. Note that even if we include the training time (15 s), it remains much lower than that of ray tracing and MWM. More importantly, as pointed out earlier and can be seen in Fig. 13(b), due to the new weights Θ', the predicted RSS values are different for the entire grid, and not just for the few measurement points. For instance, the RSS was underestimated for the entire outdoor area on the left side of the building, but after the computation of Θ' the signal for the entire area (and not only for points 26, 27, 28, 29, and 30) assumes higher values. To further test our assumption that due to calibration the predicted RSS values for the entire grid are improved, we calibrate EM DeepRay using only a fraction of the total 36 measurement points, and measuring the error with respect to the remaining points. The results are shown in Table V, where we consider cases in which EM DeepRay is calibrated using randomly only 4, 8, 16, and 27 out of the 36 measurements, and the error metrics are estimated with respect to the remaining 32, 28, 20, and 9 measurement points, respectively. As can be seen, the predictions for the rest of the grid are improved, and the accuracy of our model is increased even using a small number of points during calibration.

VII. CONCLUSION

Wireless communication system design requires robust and expedient propagation modeling tools to ensure an optimal network performance. In this article, we introduced a generalizable and realistic ML-based radio propagation modeling framework for indoor environments. Our model exploits physics-based information extracted from the blueprint of an indoor environment to predict the PL within an indoor area of interest. Unlike previous work in the field, the predictions of our model are not restricted to indoor geometries included in the training set, but it can be used to readily predict the PL for an arbitrarily complex indoor environment. Our results indicate that our model can very well replicate the PL simulated by a ray tracer, with the distinct advantage of a considerably lower computational time. More importantly, in this work we presented a calibration method that allows our data-driven model to adjust its weights to make predictions that are closer to measured rather than synthetic data. We demonstrated that after calibration, which only takes a few seconds, our model can provide estimates of RSS that resemble actual measured data with outstanding fidelity. Our work tackles two fundamental problems of ML-based propagation modeling (generalizability and credibility), and it paves the way for the establishment of a family of completely automated ML-based propagation models that will assist the deployment of next-generation wireless networks.

REFERENCES

[1] A. Čolaković and M. Hadžialić, "Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues," *Comput. Netw.*, vol. 144, pp. 17–39, Oct. 2018.

[2] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1617–1655, 3rd Quart., 2016.

[3] T. S. Rappaport *et al.*, "Wireless communications and applications above 100 GHz: Opportunities and challenges for 6G and beyond," *IEEE Access*, vol. 7, pp. 78729–78757, 2019.

[4] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[5] B. Ma, W. Guo, and J. Zhang, "A survey of online data-driven proactive 5G network optimisation using machine learning," *IEEE Access*, vol. 8, pp. 35606–35637, 2020.

[6] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3072–3108, 4th Quart., 2019.

[7] J. Wang, C. Jiang, H. Zhang, Y. Ren, K.-C. Chen, and L. Hanzo, "Thirty years of machine learning: The road to Pareto-optimal wireless networks," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1472–1514, 3rd Quart., 2020.

[8] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, 4th Quart., 2019.

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

[10] A. Seretis and C. D. Sarris, "An overview of machine learning techniques for radiowave propagation modeling," 2021, *arXiv:2101.11760*.

[11] J. Zhang, L. Liu, Y. Fan, L. Zhuang, T. Zhou, and Z. Piao, "Wireless channel propagation scenarios identification: A perspective of machine learning," *IEEE Access*, vol. 8, pp. 47797–47806, 2020.

[12] E. Damosso *et al.*, "COST action 231: Digital mobile radio towards future generation systems: Final report," Eur. Commission, Brussels, Belgium, Tech. Rep. Final Report, 1999.

[13] R. Janaswamy, *Radiowave Propagation and Smart Antennas for Wireless Communications*. Berlin, Germany: Springer, 2001.

[14] T. Rappaport, Y. Xing, G. R. MacCartney, A. F. Molisch, E. Mellios, and J. Zhang, "Overview of millimeter wave communications for fifth-generation (5G) wireless networks-with a focus on propagation models," *IEEE Trans. Antennas Propag.*, vol. 65, no. 12, pp. 6213–6230, Dec. 2017.

[15] V. Abhayawardhana, I. Wassell, D. Crosby, M. Sellars, and M. Brown, "Comparison of empirical propagation path loss models for fixed wireless access systems," in *Proc. IEEE 61st Veh. Technol. Conf.*, vol. 1, May/Jun. 2005, pp. 73–77.

[16] T. K. Sarkar, Z. Ji, K. Kim, A. Medouri, and M. Salazar-Palma, "A survey of various propagation models for mobile communication," *IEEE Antennas Propag. Mag.*, vol. 45, no. 3, pp. 51–82, Jun. 2003.

[17] D. He, B. Ai, K. Guan, L. Wang, Z. Zhong, and T. Kürner, "The design and applications of high-performance ray-tracing simulation platform for 5G and beyond wireless communications: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 10–27, 1st Quart., 2019.

[18] *Indoor 5G Networks [White Paper]*, Huawei-Carrier, Longgang Shenzhen, China, 2018.

[19] L. Azpilicueta, M. Rawat, K. Rawat, F. Ghannouchi, and F. Falcone, "A ray launching-neural network approach for radio wave propagation analysis in complex indoor environments," *IEEE Trans. Antennas Propag.*, vol. 62, no. 5, pp. 2777–2786, May 2014.

[20] X. Zhang, X. Shu, B. Zhang, J. Ren, L. Zhou, and X. Chen, "Cellular network radio propagation modeling with deep convolutional neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 2378–2386.

[21] O. Perrault, J.-P. Rossi, and T. Balandier, "Predicting field strength with a neural ray-tracing model," in *Proc. IEEE Global Telecommun. Conf.*, vol. 2, Nov. 1996, pp. 1167–1171.

[22] S. P. Sotiroudis, S. K. Goudos, and K. Siakavara, "Deep learning for radio propagation: Using image-driven regression to estimate path loss in urban areas," *ICT Exp.*, vol. 6, no. 3, pp. 160–165, Sep. 2020.

[23] S. P. Sotiroudis, P. Sarigiannidis, S. K. Goudos, and K. Siakavara, "Fusing diverse input modalities for path loss prediction: A deep learning approach," *IEEE Access*, vol. 9, pp. 30441–30451, 2021.

[24] V. V. Ratnam *et al.*, "FadeNet: Deep learning-based mm-wave large-scale channel fading prediction and its applications," *IEEE Access*, vol. 9, pp. 3278–3290, 2021.

[25] L. Wu *et al.*, "Artificial neural network based path loss prediction for wireless communication network," *IEEE Access*, vol. 8, pp. 199523–199538, 2020.

[26] J. Thrane, D. Zibar, and H. L. Christiansen, "Model-aided deep learning method for path loss prediction in mobile communication systems at 2.6 GHz," *IEEE Access*, vol. 8, pp. 7925–7936, 2020.

[27] S. P. Sotiroudis, S. K. Goudos, and K. Siakavara, "Feature importances: A tool to explain radio propagation and reduce model complexity," *Telecom*, vol. 1, no. 2, pp. 114–125, 2020.

[28] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1, no. 2. Cambridge, MA, USA: MIT Press, 2016.

[29] S. Bakirtzis, K. Qiu, J. Zhang, and I. Wassell, "DeepRay: Deep learning meets ray-tracing," in *Proc. 16th Eur. Conf. Antennas Propag. (EuCAP)*, 2022, pp. 1–5.

[30] L. Bai *et al.*, "Predicting wireless mmWave massive MIMO channel characteristics using machine learning algorithms," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–12, Aug. 2018.

[31] S. Wang *et al.*, "U-Net using stacked dilated convolutions for medical image segmentation," 2020, *arXiv:2004.03466*.

[32] I. Popescu, D. Nikitopoulos, I. Nafornita, and P. Constantinou, "ANN prediction models for indoor environment," in *Proc. IEEE Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Jun. 2006, pp. 366–371.

[33] S. Bakirtzis, T. Hashimoto, and C. D. Sarris, "FDTD-based diffuse scattering and transmission models for ray tracing of millimeter-wave communication systems," *IEEE Trans. Antennas Propag.*, vol. 69, no. 6, pp. 3389–3398, Jun. 2021.

[34] ITU-Rec, *Recommendation P.2040-1, Effects of Building Materials and Structures on Radiowave Propagation Above About 100 MHz*, International Telecommunication Union, Geneva, Switzerland, 2015.

[35] (2021). *Ranplan Wireless, Ranplan Professional*. [Online]. Available: https://ranplanwireless.com/

[36] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[37] Z. Zeng, X. Li, Y. K. Yu, and C.-W. Fu, "Deep floor plan recognition using a multi-task network with room-boundary-guided attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9096–9104.

[38] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.

[39] S. Hosseinzadeh, H. Larijani, K. Curtis, A. Wixted, and A. Amini, "Empirical propagation performance evaluation of LoRa for indoor environment," in *Proc. IEEE 15th Int. Conf. Ind. Informat. (INDIN)*, Jul. 2017, pp. 26–31.

[40] S. Hosseinzadeh. (2021). *Multi Wall (COST231) Signal Propagation Models + Python Code*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/61340-multi-wall-cost231-signal-propagation-models-python-code

Stefanos Bakirtzis (Member, IEEE) received the Diploma degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2017, and the M.A.Sc. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, University of Cambridge, Cambridge, U.K.

He is also working on the Big Data Analytics for Radio Access Networks (BANYAN) Project as a member of Ranplan Wireless Network Design Ltd., Cambridge, with the University of Cambridge. His research interests include wireless communication systems, network optimization, wireless channel modeling, machine learning and artificial intelligence, computational modeling, and stochastic uncertainty quantification.

Mr. Bakirtzis received the Onassis Foundation Scholarship and the Marie Skłodowska-Curie Actions-Innovative Training Networks (MSCA-ITN) Fellowship.

Jiming Chen received the M.S. and Ph.D. degrees in communication and information system from the University of Electronic and Science Technology, Chengdu, China, in 2003 and 2006, respectively.

He was a Research Scientist at Bell Labs, Alcatel-Lucent Shanghai Bell Ltd., Shanghai, China, from 2006 to 2011, and then became a member of the Alcatel-Lucent Technical Academy in 2010, where his research interests include the key techniques of cooperative processing and small cells for 3GPP LTE-A system. He has been the Head of Wireless Network Team and a Senior Research Fellow with Ranplan Wireless Network Design Ltd., Cambridge, U.K., since 2011, where he has also been a Principal Wireless Engineer and the Manager of the Wireless Research Group, since 2018. His research interests include key technologies for 5G NR and Wi-Fi 7, such as 5G and artificial intelligence, mm-wave communications, massive MIMO, and beamforming, for indoor and outdoor scenarios, and on the implementation of these technologies in Ranplan Wireless planning and optimization tools. Up to now, he held about more than 30 international patents and more than 30 publications.

Kehai Qiu (Student Member, IEEE) received the B.Eng. degree from the Beijing University of Posts and telecommunications, Beijing, China, in 2015, and the M.Sc. degree from The University of Sheffield, Sheffield, U.K., in 2020. He is currently pursuing the Ph.D. degree with the Computer Laboratory, University of Cambridge, Cambridge, U.K.

He is also a Doctoral Research Fellow with Ranplan Wireless Network Design Ltd., Cambridge, where he is working on the Big Data Analytics for Radio Access Networks (BANYAN) Project.

Mr. Qiu received the Marie Skłodowska-Curie Actions-Innovative Training Networks (MSCA-ITN) Fellowship.

Jie Zhang (Senior Member, IEEE) received the Ph.D. degree from the East China University of Science and Technology, Shanghai, China, in 1995.

He has been the Chair in wireless systems with the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, U.K., since January 2011. He is also the Founder, the Board Director, and the Chief Scientific Officer of Ranplan Wireless, Cambridge, U.K., a public company listed on Nasdaq First North. Ranplan Wireless produces a suite of world leading indoor and the first joint indoor-outdoor radio access network planning tool–Ranplan Professional, which is being used by the world's largest mobile operators and network vendors across the globe. Along with his students and colleagues, he has pioneered research in small cell and heterogeneous network and published some of the landmark papers and book on these topics, widely used by both academia and industry. Since 2010, he and his team have also developed ground-breaking work in building wireless performance modeling, evaluation, and optimization, the key concepts of which were introduced in "Fundamental Wireless Performance of a Building," IEEE Wireless Communications, 29(1), 2022.

Ian Wassell received the B.Sc. and B.Eng. degrees from the University of Loughborough, Loughborough, U.K., in 1983, and the Ph.D. degree from the University of Southampton, Southampton, U.K., in 1990.

He is currently a University Associate Professor with the Computer Laboratory, University of Cambridge, Cambridge, U.K., and has experience in excess of 25 years in simulation and design of radio communication systems gained via a number of positions in industry and higher education. He has authored more than 200 articles. His current research interests include wireless sensor networks, cooperative wireless networks, propagation modeling, sparse representation, and machine learning.

Dr. Wassell is a member of the IET and a Chartered Engineer.