IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 5, 2020, accepted October 13, 2020, date of publication October 22, 2020, date of current version November 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3032933

Convolutional-Type Neural Networks for Fading Channel Forecasting

LIA AHRENS, JULIAN AHRENS, AND HANS DIETER SCHOTTEN, (Member, IEEE)
German Research Center for Artificial Intelligence (DFKI), 67663 Kaiserslautern, Germany

Corresponding author: Lia Ahrens (lia.ahrens@dfki.de)

This work was supported by the Federal Ministry of Education and Research of the Federal Republic of Germany through the FunKI Project under Grant 16KIS1182.

ABSTRACT In this article, a series of convolutional-type predictive neural networks are proposed for the issue of fading channel forecasting for orthogonal frequency-division multiplexing (OFDM) transmission systems in a multiple-input and multiple-output (MIMO) mode via a noisy channel. The proposed neural networks all employ convolutional connections that operate in a translation-invariant manner in the frequency domain of the time-varying channel transfer function, which effectively tackles the essential challenges of high dimensionality and denoising. Each of the proposed convolutional-type neural networks is built on a specific overall network architecture and functions as an independent predictor that offers advantages regarding a specific aspect such as accuracy over a certain prediction span or computational effort. Comparative evaluations against common prediction methods such as the Kalman filtering scheme and the standard long-short term memory units (LSTMs) are provided on the basis of transmission simulations over dispersive fading channels with Rayleigh components according to the well-established 3GPP Long Term Evolution (LTE) standards.

INDEX TERMS Wireless communications, deep learning, convolutional neural networks, LSTMs, time series analysis, fading channel forecasting, MIMO.

I. INTRODUCTION

Dynamic radio resource management and adaptive modulation and coding schemes are essential parts of modern cellular networks. In order to maximise spectral efficiency, it is necessary for these schemes to be adapted to the current channel transmission properties. Since scheduling and coding have to be performed before the actual transmission, it is necessary to anticipate the channel quality ahead of time. In the presence of multipath propagation and moving receivers, time variance in the channel transfer function and frequency selective fading significantly increase the complexity of the issue of channel quality forecasting which requires a powerful predictive model.

The central objective of this article is to predict future states of a noise-corrupted fading channel under certain propagation conditions for transmitting orthogonal frequency-division multiplexing (OFDM) symbols, operating in a multiple-input and multiple-output (MIMO) transmission mode. A big challenge when forecasting noisy time series in such a context consists in the high dimensionality issue arising in the multi-subcarrier setting. This work aims to design predictive models that are parsimonious but still complex enough to capture the essential characteristics of such high-dimensional noisy time series.

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Asif.

A. BACKGROUND AND RELATED WORK

From a mathematical point of view, the vast majority of previous contributions to fading channel forecasting employ stochastic methods or machine learning techniques. The mathematical ingredients of these techniques and their relationships are briefly summarised below, followed by their applications to wireless communications.

Most of the classical approaches to time series forecasting arise from the general framework of stochastic filtering in a state-space setting. Here, the conditional distribution of a Markovian latent variable, called hidden state, is estimated and used as the key object to relate the past and current observations to the best future forecasts [1]. Among the general state-space models, the theoretically best-understood case in terms of explicit optimal estimator

and asymptotic behaviour of suboptimal solutions is the linear setting with Gaussian white noise, for which the well-known Kalman filter (KF) [1], [2] operating as a recursive predictor is derived. Another common prediction scheme from the field of parametric statistics is provided by the autoregressive integrated moving average (ARIMA) methodology which is based on spectral theory [3]. In particular, the autoregressive (AR) model for stationary time series is an instance of ARIMA models. Each ARIMA model with Gaussian disturbance has an equivalent linear state-space model representation with associated Kalman filtering scheme. In the context of fading channel prediction, the application of AR models with associated Kalman filtering scheme has been well studied, both for flat fading scenarios, e.g. by Duel-Hallen [4] and Duel-Hallen *et al.* [5], and for frequency-selective fading in OFDM systems, e.g. Schafhuber and Matz (2005) [6], as well as in noisy channels, e.g. Sharma and Chandra (2007) [7]. Stochastic filtering approaches are powerful and noise resistant, provided that the mathematical modelling is reasonable. However, when applying the most common of these techniques, the KF, the underlying linear model assumption restricts its capability of forecasting time series of more complex nature.

Apart from state-space models, more specialised statistical models such as sinusoidal models and corresponding spectral algorithms have also been employed for the task of fading channel prediction through spectral methods as in Zoltowski, Haardt, and Mathews (1996) [8]. While these methods produce impressive results when evaluated on comparatively simple simulated radio channels such as those devised from the classical Jakes model (1974) [9], Semmelrodt and Kattenbach (2003) [10] found that they appear less powerful when faced with more complicated radio channel models or real-world physical measurements.

In recent years, with the massive increase of available computational power, machine learning approaches making use of predictive neural networks have seen increased popularity in time series forecasting. In general, machine learning techniques aim to extract essential information from complex and extensive datasets. Such an information extraction task can be formulated as a function estimation problem which is studied more theoretically in non-parametric statistics (e.g. [11]). Neural networks are nonlinear systems with intermediate steps (layers) that combine affine transformations (inter-layer connections) and fixed nonlinearities (activation functions), where the coefficients of the affine transformations serve as model parameters that are to be fitted through training. The architecture of most neural networks can be classified according to the direction of data flow as either recurrent or feedforward. The most representative recurrent predictive neural networks are the standard recurrent neural networks (RNNs) as introduced in Elman (1990) [12] and the standard long-short term memory units (LSTMs) by Hochreiter and Schmidhuber (1997) [13], both employing full connections. Functioning as recursive predictors, recurrent neural networks resemble the stochastic filtering schemes in the sense that they also incorporate both the ongoing observations and the updated hidden states into the future forecasts. Among the feedforward neural networks, convolutional neural networks (CNNs), e.g. as introduced in [14], provide a powerful class of predictors, e.g. van den Oord *et al.* (2016) [15]. The core components of CNNs are the so-called convolutional layers which operate as tunable finite impulse response filters followed by fixed nonlinearities in a translation-invariant manner. Functioning as predictors, CNNs resemble the finite-order AR models in the sense that the forecasts depend on the temporal evolution of a fixed number of observations from the past (local receptive field in time). In wireless communications, machine learning methods employing neural networks have been applied to a wide range of problems related to the physical layer of radio transmission [16], [17], including transmitter and receiver design, e.g. Aoudia and Hoydis (2018) [18], Felix *et al.* (2018) [19], and fading channel modelling, e.g. Ye *et al.* (2018) [20]. On the issue of fading channel prediction, standard predictive neural networks with full connections have been evaluated in various contexts in single-subcarrier settings, e.g. Ding and Hirose (2014) [21], Liao *et al.* (2018) [22], Jiang and Schotten (2019, 2020) [23], [24], and Yuan *et al.* (2020) [25], with [25] employing an extra CNN channel classifier to identify patterns in the autocorrelation function of the channel prior to applying the actual predictor. Overall, while the Kalman filtering scheme is based on linear models and parametrised probability assumptions, predictive neural networks incorporate nonlinearities and no specific interpretation (such as Kalman gain or conditional state variance) is imposed on their model parameters. In fact, the tunable parameters of a neural network only provide the means for general function approximation. Therefore, more complex mathematical models may emerge.

B. CONCEPT OF THIS PAPER

In this work, a series of non-standard convolutional-type predictive neural networks are proposed for multi-step ahead fading channel forecasting in wireless communications. In contrast to the majority of previous contributions that focused on single-subcarrier settings, this work considers the more complex multi-subcarrier setting and provides a solution to the associated high dimensionality problem. As opposed to [6] where a multi-subcarrier setting was considered and the actual channel forecasting is conducted in the time domain after inverse Fourier transformation of the channel transfer functions, the predictors proposed herein work directly in the frequency domain where the channel state information is processed through shift-invariant convolutional layers. This approach is motivated by the following consideration:

In [6], it is shown that when forecasting the impulse response of a fading channel in the time domain via a linear minimum mean square error predictor, the values of the impulse response for each of the different delay times can be forecast separately, as they are pairwise uncorrelated

in a wide-sense stationary uncorrelated scattering channel. However, separate predictors with separate model parameters have to be used for each different delay time. In contrast, when working in the frequency domain, considering the correlation between adjacent subcarriers, an accurate prediction of the transfer function at a single subcarrier requires the consideration of its neighbouring subcarriers, which increases the complexity of the prediction. However, when considering common mobile communication channels, the utilised bandwidth is always significantly smaller than the carrier frequency, which leads to the time varying transfer function behaving very similarly across the entire bandwidth. This suggests the feasibility of working in the frequency domain and choosing a shift-invariant predictive model that applies to all subcarriers. As introduced in Section I-A, when applied to the frequency axis of the channel state information, the convolution operation as used in CNNs provides a suitable component for devising such a shift-invariant predictor.

In this work, not only one but multiple variants of such a convolutional-type predictor are designed. The reasons for this are as follows: First, when producing multi-step ahead predictions, the performance of a neural network varies over the total prediction span, depending on the overall neural network architecture (e.g. recurrent or feedforward). According to previous observations, e.g. as remarked by Mathieu *et al.* (2016) [26], CNN predictors (feedforward) excel in long-term forecasting, whereas LSTM predictors (recurrent) are more favourable in short-term forecasting. Moreover, experimental results show that manipulating the receptive field of a predictive neural network may also have an impact on the performance. Accordingly, four classes of convolutional-type predictors are designed herein, with distinctive characteristics of each stated in the following:

Partially dilated CNN predictor

The frequency and time axes of the channel state information are processed jointly through two-dimensional convolutional layers, where dilation factors are applied to the time axis so as to incorporate long-term dependency in time.

The current version of this type of predictor generalises the original one introduced in Ahrens *et al.* (2019) [27] to a MIMO setting and uses extra skip connections to achieve performance improvement.

Convolutional LSTM predictor

The frequency and time axes of the channel state information are processed separately through one-dimensional convolutional layers and LSTM units, respectively. Basically, one-dimensional convolutional layers are encapsulated in an LSTM architecture.

F_0-convolutional LSTM predictor

The original network architecture of the convolutional LSTM predictor is augmented with an extra initial transformation that partitions the observation series of channel states into overlapping segments, each covering a pre-determined amount of subcarriers. In doing so, the size of the receptive field in the frequency domain is made a tunable hyperparameter of the corresponding predictor.

Combined neural network predictor

A final convolutional layer is attached to the best long-term predictor (partially dilated CNN) and the best short-term predictor (F_0-convolutional LSTM). By training this extra final layer, the overall performance is improved.

Among these four classes of predictors, the first two are computationally most efficient and the last three are introduced in order of ascending prediction accuracy. Such progressive performance improvement, however, comes at the price of additional computational complexity. The two aspects of performance and computational effort are considered separately in the evaluation.

To the best of the authors' knowledge, this is the first time that such an extensive variety of specifically designed convolutional-type predictive neural networks are applied to the task of fading channel forecasting in wireless communications. All of the proposed predictors share the following advantages over previously proposed methods:

- While most of the previous contributions to fading channel forecasting focused on single-subcarrier settings, the proposed convolutional-type predictive neural networks apply to the more complex multi-subcarrier setting (OFDM transmission).
- Compared to existing methods employing the KF for AR model, standard RNN, or LSTM, including both the direct adaption of previous single-subcarrier predictors to the present setting and the time-domain approach in [6] for OFDM transmission, the proposed predictors deliver significantly better performance. This in particular shows the benefit of working in the frequency domain and incorporating non-trivial receptive fields into the predictor in a multi-subcarrier setting.
- Even though previous single-subcarrier predictors can be made more powerful by incorporating multiple adjacent subcarriers as extra dimensions into the input time series, the computational complexity of such an approach is significantly higher than that of the proposed predictors. This in particular demonstrates the computational efficiency of employing convolutional layers for processing high dimensional channel transfer functions.
- Traditional multi-step ahead prediction schemes rely on iterative one-step ahead prediction, which is tedious and susceptible to inherited errors. In contrast, the proposed predictors deliver future channel states over the entire horizon simultaneously, with higher prediction accuracy. Here, each prediction time step is treated as a separate channel in the output layer so that prediction errors over the entire horizon are incorporated concurrently into the loss function and minimised jointly through training.
- Although this work only addresses some of the issues encountered in more complex MIMO settings,

the experiments performed herein reveal a non-obvious finding concerning neural network architectures: In a MIMO setting, it is more effective to initially keep the channel state information related to different transmitter-receiver antenna pairs separate and only combine the intermediate results in a final layer rather than allowing inter-antenna connections throughout the entire neural network.

The remainder of the paper is organised as follows. The aforementioned four families of predictive neural networks designed for the fading channel prediction task are presented in Section II, followed by their common training algorithm using supervised learning described in Section III. Section IV introduces the evaluation datasets which are generated via simulated channel transmission. A detailed specification of the data generation scheme is provided as supplementary material. In Section V, the proposed predictive neural networks are evaluated and compared with some of the common reference predictors whose descriptions are provided in Appendix A. As an outlook, a possible way to extend the current approach to a setting with time-variant channel conditions is discussed in Section VI. Section VII concludes the paper.

II. CHANNEL PREDICTION MODELS

Henceforth, let F denote the total number of subcarriers for which channel state information is available and let P and Q denote the total number of transmitter antenna ports and receiver antennae, respectively. For the multi-step ahead prediction task, let R denote the total number of desired prediction time steps (total prediction span). One of the most useful properties of a properly designed OFDM transmission system is that the impact of the channel on the signal can be characterised by a $\mathbb{C}^{(P\times Q)\times F}$-valued time-series, $H^{\text{true}} = \{H_t^{\text{true}}\}_t$ with $H_t^{\text{true}} = \{H_t^{\text{true}\,(p,q)}[f]\}_{(p,q),f} \in \mathbb{C}^{(P\times Q)\times F}$ for all t, called the *time-varying channel transfer function* or the *channel state information*. As shown by Peled and Ruiz (1980) [28], this characterisation is possible even in the case of a multipath fading channel as long as the cyclic prefix used in the OFDM transmission is longer than the maximum delay spread of the channel. In practice, it is impossible to exactly measure the actual channel transfer function. However, using reference signals embedded in the transmitted signal as specified in 3GPP TS 36.211 [29], the channel transfer function can be estimated from the received signal. In the sequel, the estimated channel transfer function (observation series), which is a slightly distorted and noise-corrupted approximation of the actual channel transfer function, will be denoted by $H^{\text{est}} = \{H_t^{\text{est}}\}_t$ with $H_t^{\text{est}} = \{H_t^{\text{est}\,(p,q)}[f]\}_{(p,q),f} \in \mathbb{C}^{(P\times Q)\times F}$ for all t. The resolution of the time series H^{est} and H^{true} used in this specific exposition of the method is given in Section IV.

The channel forecasting problem consists in finding an appropriate mathematical model, Γ, called *predictor*, such that for any t, given observations $\{H_{t'}^{\text{est}}\}_{t'\le t}$ up to time t, $\Gamma(\{H_{t'}^{\text{est}}\}_{t'\le t})$ delivers a reasonable forecast of the actual channel transfer function up to R steps ahead, $\{H_{t+r}^{\text{true}}\}_{r=1,\dots,R}$. In this section, a variety of predictors employing predictive neural networks are derived for the task of multi-step ahead channel prediction. Each predictive neural network represents a family of predictors $\Gamma(\cdot,\boldsymbol{\xi})$ parametrised by a real-valued vector of parameters $\boldsymbol{\xi} \in \mathbb{R}^{\Xi}$.

Since the neural networks employed herein use real-valued tensors as their inputs and outputs, data transformations that convert the complex-valued channel state information into real-valued tensors and vice versa are required before and after invoking the neural network, respectively. The notion of *tensor* used in this work is that of an indexed family $x = \{x_{i_0},\dots,x_{i_{d-1}}\}_{i_0\in I_0,\dots,i_{d-1}\in I_{d-1}}$ where the index sets I_j are finite sets of integers. The total number of indices, d, is referred to as the *dimension* of the tensor x and the positions of the indices, $0,\dots,d-1$, are referred to as the *axes* of the tensor x. In this article, all of the proposed neural networks have in common that they first process the channel state information related to different pairs of antennas $(p,q) \in P\times Q$ separately and combine the intermediate information in the final layer of the neural network. Accordingly, the observation series H^{est} is first converted into $(P\cdot Q)$-many three-dimensional tensors $x^{\text{in}\,(p,q)}$ by setting

$$\begin{aligned} x_{t,0,f}^{\text{in}\,(p,q)} &:= \operatorname{Re} H_t^{\text{est}\,(p,q)}[f] \\ x_{t,1,f}^{\text{in}\,(p,q)} &:= \operatorname{Im} H_t^{\text{est}\,(p,q)}[f] \end{aligned} \tag{1}$$

for all p,q,t, and f. The tensors $x^{\text{in}\,(p,q)}$ for all pairs (p,q) form the input of the neural network. The output of the predictive neural network is again a three-dimensional tensor y^{out} of the form $\{y_{t,\lambda,f}^{\text{out}}\}_{t,\lambda,f}$, with the index λ combining the antenna pair indices (p,q), a real or imaginary part flag, and the prediction step r. More precisely, the value

$$\begin{aligned} y_{t,((p\cdot Q+q)\cdot 2+0)\cdot R+r-1,f}^{\text{out}} + i\,y_{t,((p\cdot Q+q)\cdot 2+1)\cdot R+r-1,f}^{\text{out}} \\ =:\ \Gamma(\{H_{t'}^{\text{est}}\}_{t'\le t},\boldsymbol{\xi})_r^{(p,q)}[f] \end{aligned} \tag{2}$$

is the prediction for $H_{t+r}^{\text{true}\,(p,q)}[f]$ available at time t. (Here, i denotes the imaginary unit.) In the subsequent presentation of the proposed neural networks, all operations are expressed in terms of tensor transformations and each neural network layout describes the transformation from the input tensors $x^{\text{in}\,(p,q)}$ for all (p,q) to the output tensor y^{out}.

A. d-DIMENSIONAL CONVOLUTIONAL LAYERS

The cornerstone of the predictive neural networks proposed in this work is the d-dimensional *convolutional layer* which in turn is based on the d-dimensional convolution operation. To simplify notation so that there is no need to distinguish between different dimensions, in the sequel, multi-index notation will be used, i.e., for a d-dimensional tensor $x = \{x_{u_0,\dots,u_{d-1}}\}_{u_0,\dots,u_{d-1}}$, its indices $u_0,\dots,u_{d-1}$ will be regarded as a vector $\boldsymbol{u} = (u_0,\dots,u_{d-1})$ and its entries will be referred to as $x_{\boldsymbol{u}}$. Let now $d \in \mathbb{N}$ be a fixed dimension and let $x = \{x_{u_0,\dots,u_{d-1}}\}_{u_0,\dots,u_{d-1}} = \{x_{\boldsymbol{u}}\}_{\boldsymbol{u}}$ and $k = \{k_{v_0,\dots,v_{d-1}}\}_{v_0,\dots,v_{d-1}} = \{k_{\boldsymbol{v}}\}_{\boldsymbol{v}}$ be d-dimensional tensors. The d-dimensional *convolution* of x with k, denoted

by $x * k = \{(x * k)_{\boldsymbol{u}}\}_{\boldsymbol{u}}$, is defined as

$$(x * k)_{\boldsymbol{u}} := \sum_{\boldsymbol{v}} x_{\boldsymbol{u}-\boldsymbol{v}} k_{\boldsymbol{v}} \tag{3}$$

for all multi-indices $\boldsymbol{u}$, where the sum is formed over all multi-indices $\boldsymbol{v}$ for which the terms $x_{\boldsymbol{u}-\boldsymbol{v}}$ and $k_{\boldsymbol{v}}$ are defined. When performed in the convolutional layer of a neural network, the tensor k consists of tunable parameters of the neural network and is called the *convolution kernel*. Note that, while the right-hand side of (3) is technically defined for all multi-indices $\boldsymbol{u}$, it can only be of non-zero value if there exists at least one multi-index $\boldsymbol{v}$ in the index set of the convolution kernel k such that $\boldsymbol{u}-\boldsymbol{v}$ lies within the index set of the tensor x. In the following, it will always be assumed that $\mathbf{0} = (0, \ldots, 0)$ is contained in the index set of the convolution kernel and the result of the convolution operation $x * k$ will be restricted to the index set of the tensor x.

A d-dimensional *convolutional layer* maps a $(d+1)$-dimensional input tensor x to a $(d+1)$-dimensional output tensor y. The first index of the tensors x and y is referred to as the *channel index*. Correspondingly, the sizes of the corresponding index sets are referred to as the *total number of input channels* and the *total number of output channels* of the convolutional layer. The transformation performed by the convolutional layer consists of an affine transformation called the *convolutional connections*, followed by a fixed and typically nonlinear *activation function*. More precisely, the convolutional connections map the input tensor x to an intermediate tensor y^{int} via

$$y^{\text{int}}_{j,\star} = \sum_{i} x_{i,\star} * k_{i,j,\star} + b_j \tag{4}$$

where k is a $(d+2)$-dimensional tensor used as a matrix of d-dimensional *convolution kernels* and $b = \{b_j\}_j$ is a *bias vector*. Together, k and b form the *free parameters* of the convolutional layer. In (4), the convolution operations are performed along the d-dimensional multi-indices in the position of $\star$ and the indices i and j range over all input and output channel indices of the convolutional layer, respectively. The output tensor y is then obtained by applying the activation function α of the convolutional layer to each of the entries of the intermediate tensor y^{int}, i.e.,

$$y_{j,\boldsymbol{u}} := \alpha(y^{\text{int}}_{j,\boldsymbol{u}}) \tag{5}$$

for all output channel indices j and all multi-indices $\boldsymbol{u}$.

1) CAUSALITY

In the case where one of the axes of the tensor x represents time, the operation of convolving with a convolution kernel k is a causal operation if the index set of k that corresponds to the temporal axis consists only of non-negative integers. In other words, if all valid temporal indices of k are non-negative, the value of $x * k$ at time step t depends only on the values of x up to time t. As the aim of this work is to design a predictor, it is important that all operations used in the prediction scheme are causal. Accordingly, temporal indices of convolution kernels will always be chosen to consist purely of non-negative integers in order to ensure causality of the convolution operation. Frequency axis indices will always be chosen as symmetrical ranges, i.e., ranges of the form $\{-n, \ldots, n\}$.

2) TRANSLATION INVARIANCE AND LOCAL RECEPTIVE FIELD

Two important properties of convolutional layers can be inferred from equations (3), (4), and (5), which are essential to the design of the proposed predictors.

First, convolutional layers are *translation invariant*, i.e., if any of the non-channel indices of the input tensor x are shifted by fixed offsets, this will result in the same shifts appearing in the output tensor y.

Second, each convolutional layer has a *local receptive field*, i.e., the value of each entry of the output tensor y depends only on a finite number of entries of the input tensor x located at fixed offsets from the index of the output tensor entry. More precisely, if V is the finite set of multi-indices such that the convolution kernels $k_{i,j,*}$ are defined on $-V$, then, for a given multi-index $\boldsymbol{u}$, the value $y_{j,\boldsymbol{u}}$ depends only on the values $x_{i,\boldsymbol{u}'}$ with $\boldsymbol{u}' - \boldsymbol{u} \in V$ and the convolutional layer is said to have *local receptive field* V. Observe that, if L-many convolutional layers with local receptive fields $V^{(0)}, \ldots, V^{(L-1)}$ are stacked, i.e. applied in succession, the resulting total transformation has a cumulative local receptive field of $V^{(0)} + \cdots + V^{(L-1)}$, that is, the value of the transformed tensor $y_{j,\boldsymbol{u}}$ depends on the value of the input tensor $x_{i,\boldsymbol{u}'}$ only if there exist offsets $\boldsymbol{v}^{(0)} \in V^{(0)}, \ldots, \boldsymbol{v}^{(L-1)} \in V^{(L-1)}$ such that $\boldsymbol{u}' - \boldsymbol{u} = \boldsymbol{v}^{(0)} + \cdots + \boldsymbol{v}^{(l-1)}$. As the cumulative local receptive field determines the amount of historical information available for forecasting, it is a key characteristic of a predictive neural network and will be discussed in more detail for each of the predictors introduced below.

B. PARTIALLY DILATED CNN PREDICTOR

The notion of a *partially dilated convolutional neural network* (CNN) was originally introduced in [27] for multi-step channel forecasting in a single-input and single-output (SISO) setting under a different environmental condition, incorporating some of the ideas from [15]. The layout of the CNN presented in this section is designed for the more challenging general MIMO setting and can be considered as a refined version of that in [27] in the sense that performance is improved by incorporating skip connections that link all hidden layers directly to the output layer.

Most CNNs employ convolution kernels that are indexed by ranges of integers with step size one, usually of the form $\{0, \ldots, n-1\}$. By replacing these ranges with index sets of the form $\{0, \delta, 2\delta, \ldots, (n-1)\delta\}$, where $\delta \in \mathbb{Z}_{\geq 1}$ is an additional hyperparameter called the *dilation factor*, the actual convolution kernel is effectively spread out (dilated) along the corresponding axis without altering the number of free parameters. From (3) (note in particular, how the multi-index $\boldsymbol{v}$ affects the weighting of the tensor x) and the discussion on local receptive fields in Section II-A.2, incorporating dilation

into the convolution kernels of a CNN drastically increases the size of its local receptive field. For the task of multi-step ahead channel prediction, the proposed partially dilated CNN encapsulates two-dimensional convolutional layers that operate simultaneously on the temporal and frequency axes of the observation series H^{est}, that is, in (4), let $d = 2$ and let $\star$ represent the t- and f-axes, with dilation factors applied to the t-axis.

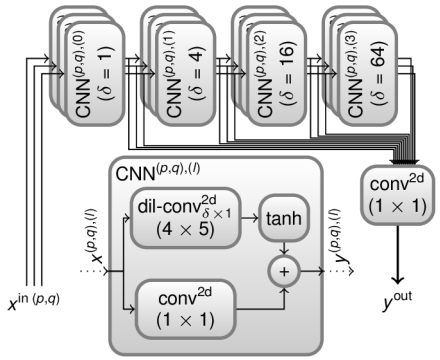

FIGURE 1. Layout of partially dilated CNN predictor.

1) NETWORK LAYOUT OF PARTIALLY DILATED CNN PREDICTOR

The network layout of the proposed partially dilated CNN predictor is displayed in FIGURE 1, with the corresponding data flow described in the following. First of all, since convolutional layers assume the first index of their input tensors to be the channel index (recall Section II-A), the first two indices of the tensors $\{x^{\text{in}\,(p,q)}_{t,\lambda,f}\}_{t,\lambda,f}$ and $\{y^{\text{out}}_{t,\lambda,f}\}_{t,\lambda,f}$ introduced in equations (1) and (2) have to be interchanged prior to and after being processed through the CNN, respectively. The partially dilated CNN predictor employs $(P \cdot Q)$-many independent sub-CNNs that are connected in a final layer, where each sub-CNN is related to a different antenna pair (p, q) and takes the tensor $x^{\text{in}\,(p,q)}$ as input. Each antenna pair-specific sub-CNN consists of 4 consecutive residual blocks, $l = 0, 1, 2, 3$, with 2 initial input channels as determined by the shape of $x^{\text{in}\,(p,q)}$, followed by 12 output channels per block. Each residual block (labelled $\text{CNN}^{(p,q),(l)}$ in FIGURE 1) is composed of 2 initially separate two-dimensional convolutional layers (recall (4) and (5) with $d = 2$): one with kernel size 4×5, dilated by a factor of $\delta \times 1$ with $\delta = 4^l$ and using the hyperbolic tangent activation function, tanh, the other with a kernel size of 1×1 and using no activation function. The outputs of these convolutional layers are then added together. The outputs of each of the residual blocks from all of the antenna pair-specific sub-CNNs are concatenated along the channel axis into a single tensor and then processed through a final convolutional layer with $(P \cdot Q \cdot 2 \cdot R)$-many output channels and kernel size 1×1, using no activation function.

The above neural network layout, in particular the choice of layer sizes, is adapted to the evaluation datasets used in the present exposition of the method (cf. Section IV). In general, hyperparameter choice should always be adjusted to the nature of the underlying data. Note that incorporating residual blocks is a common technique used in many modern CNN architectures, which aims to improve the backpropagation of the gradient, thereby speeding up the training process as demonstrated by He *et al.* (2016) [30]. The residual connections in the present work are applied in the same manner as the ones in [15]. As a refinement over the neural network architecture proposed in [27], the introduction of the so-called *skip connections*, e.g. as introduced in Graves (2013) [31], in the current neural network, i.e., using the outputs from all the residual blocks as the input to the final layer instead of only the output from the last block of each sub-CNN, increases the prediction accuracy and further contributes to speeding up convergence [15].

Remark: A straightforward treatment of a general MIMO setting would be to employ a joint predictive neural network that takes all of the tensors $x^{\text{in}\,(p,q)}$ concatenated over all antenna-pair indices (p, q) as a whole as input. Compared to such an approach, the proposed architecture of separating the channel state information associated with different antenna pairs in the early layers turns out to be more feasible in experiments. This applies to all predictive neural networks and all types of MIMO correlation considered in this article. To some extent, the transformation performed in the final layer is trained to mimic the MIMO correlation of the underlying transmission channel. This layout aids greatly in low MIMO correlation scenarios where the connections between the different transmitter-receiver groups are expected to be weak, whereas in high MIMO correlation scenarios, only allowing inter-antenna pair connections in the final layer still proves to be more effective than allowing such connections throughout the entire neural network.

2) LOCAL RECEPTIVE FIELD OF PARTIALLY DILATED CNN PREDICTOR

As stated in Section II-A.1, the kernels of the convolutional layers are causal along the temporal axis and centred on the frequency axis. Therefore, for each residual block $l = 0, 1, 2, 3$ introduced in Section II-B.1 (recall $\text{CNN}^{(p,q),(l)}$ in FIGURE 1), the local receptive field $V^{(l)}$ (recall the definition in Section II-A.2) is given by

$$V^{(l)} = \{-3 \cdot 4^l, -2 \cdot 4^l, -4^l, 0\} \times \{-2, -1, 0, 1, 2\}.$$

The cumulative local receptive field of the partially dilated CNN predictor then amounts to

$$V^{(0)} + V^{(1)} + V^{(2)} + V^{(3)} = \{-255, \ldots, 0\} \times \{-8, \ldots, 8\}.$$

This is illustrated by the green asterisk-filled box in FIGURE 2, where the receptive fields of each of the proposed predictors are displayed superimposed onto a plot of a typical observation series H^{est}. Notice the oblong shape of the receptive field of the partially dilated CNN predictor that results from dilating only along the temporal axis.

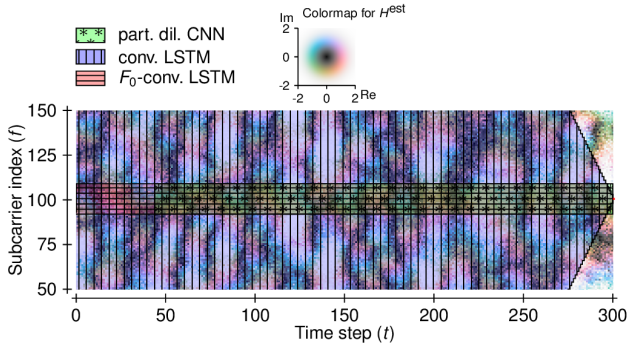

FIGURE 2. Receptive fields of different predictive neural networks.

C. CONVOLUTIONAL LSTM PREDICTOR

In contrast to the partially dilated CNN using a feedforward architecture, the *convolutional LSTM* to be introduced in this section is a recurrent predictor. An LSTM processes the input data time step by time step, retaining a *hidden state* that contains a so-called *cell value* and is carried over between time steps. The output of the current time step is computed from the current input and the previous hidden state. In a convolutional LSTM, all connections linking these values to the current output are convolutional layers that operate in a translation-invariant manner along one or more axes (e.g. Shi *et al.* (2015) [32]). In the present setting, one-dimensional convolutional layers are used to process the frequency axis of the observation series H^{est}, that is, in (4), let $d = 1$ and let $\star$ represent the f-axis.

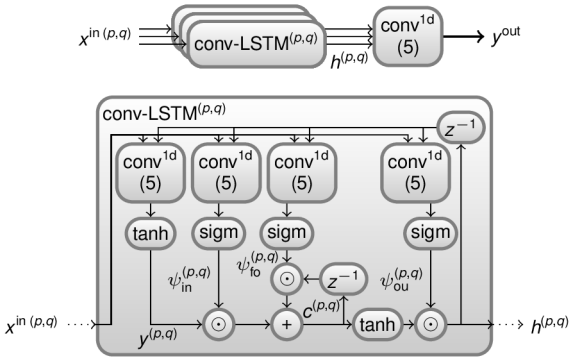

FIGURE 3. Layout of convolutional LSTM predictor.

1) NETWORK LAYOUT OF CONVOLUTIONAL LSTM PREDICTOR

The layout of the proposed convolutional LSTM predictor is illustrated in FIGURE 3, with the corresponding data flow described in the following. Like the partially dilated CNN, the convolutional LSTM predictor employs $(P \cdot Q)$-many separate subnetworks that are connected in a final convolutional layer, where each subnetwork takes one of the antenna pair-specific tensors $x^{\text{in}\,(p,q)}$ in (1) as input. The antenna pair-specific subnetworks (blocks labelled conv-LSTM$^{(p,q)}$ in FIGURE 3) are themselves convolutional LSTMs, each consisting of 4 one-dimensional convolutional layers (recall (4) and (5) with $d = 1$), 3 gates (input, forget, and output gate), 1 auxiliary activation function, and 2 feedback lines (indicated by z^{-1} in FIGURE 3). The operation of each sub-convolutional LSTM proceeds as follows: For each index pair (p, q), let $c_t^{(p,q)} = \{c_{t,\lambda,f}^{(p,q)}\}_{\lambda,f}$ and $h_t^{(p,q)} = \{h_{t,\lambda,f}^{(p,q)}\}_{\lambda,f}$ denote the tensors of cell value and remaining hidden state at time step t, respectively, which are two-dimensional tensors with 12 channels (λ-axis) and initialised to zero for $t = -1$. For each time step $t \geq 0$, the current input tensor $x_t^{\text{in}\,(p,q)}$ is concatenated with the previous hidden state $h_{t-1}^{(p,q)}$ along the channel axis, resulting in a tensor with $2 + 12$ channels. This tensor is then processed in parallel through 4 separate one-dimensional convolutional layers with 12 output channels and kernel size 5. Three of these convolutional layers are associated with gates and use the sigmoid activation function, sigm, which is defined by

$$\text{sigm}(x) := \frac{1}{\exp(-x) + 1},$$

with output tensors denoted by $\psi_{\text{in}\,t}^{(p,q)}$, $\psi_{\text{fo}\,t}^{(p,q)}$, and $\psi_{\text{ou}\,t}^{(p,q)}$ and referred to as input, forget, and output gate, respectively. The remaining convolutional layer uses the hyperbolic tangent activation function, tanh, with output tensor denoted by $y_t^{(p,q)}$. Following these 4 convolutional layers, the current cell value $c_t^{(p,q)}$ and hidden state $h_t^{(p,q)}$ are computed in a recurrent manner via

$$\begin{aligned} c_t^{(p,q)} &:= y_t^{(p,q)} \odot \psi_{\text{in}\,t}^{(p,q)} + c_{t-1}^{(p,q)} \odot \psi_{\text{fo}\,t}^{(p,q)} \\ h_t^{(p,q)} &:= \tanh(c_t^{(p,q)}) \odot \psi_{\text{ou}\,t}^{(p,q)} \end{aligned}$$

where $\odot$ denotes the Hadamard product and the additional tanh activation function is applied entry-wise as usual. Finally, the output tensors of the antenna specific-subnetworks, $h_t^{(p,q)}$ for all indices p and q, are concatenated along their channel axes and passed through a final convolutional layer with $(P \cdot Q \cdot 2 \cdot R)$-many output channels and kernel size 5, using no activation function.

As with the partially dilated CNN predictor, the above network layout, in particular the hyperparameter choice, is adapted to the evaluation datasets introduced in Section IV and should be adjusted in a data-adaptive manner in general. As noted in the remark at the end of Section II-B.1, separating the channel state information associated with different antenna pairs near the input layer of the neural network improves the performance in a general MIMO setting.

2) RECEPTIVE FIELD OF CONVOLUTIONAL LSTM PREDICTOR

Analysing the receptive field of the convolutional LSTM predictor requires unfolding the network architecture. Unfolding is basically performed by creating a countably infinite number of copies of the network, one copy per time step, and replacing each delay loop inside any of these copies by a connection to the corresponding point in the preceding copy. For a more formal description of unfolding, see Zhang *et al.* (2016) [33]. This essentially converts the

convolutional LSTM into an infinite stack of convolutional layers, each receiving an input from a different time step along with the hidden state of the preceding layer. Performing the above analysis reveals that the cumulative receptive field of the convolutional LSTM predictor is cone-shaped, expanding along the frequency axis when moving backwards into the past until the entire spectrum is covered. This is illustrated in FIGURE 2 by the blue area filled with vertical lines.

The shape of the receptive field differs significantly from that of the partially dilated CNN predictor which is much narrower in the frequency domain. In experiments, the partially dilated CNN predictor exhibits better performance than the convolutional LSTM predictor, which suggests that localising the receptive field in the frequency domain may boost performance. In the following section, a modification of the current convolutional LSTM predictor will be devised, where the size of the receptive field in the frequency domain is reduced to that of the partially dilated CNN predictor.

D. F_0-CONVOLUTIONAL LSTM PREDICTOR

Based on the discussion in Section II-C.2 on the receptive field of the preceding predictors, a modified version of the convolutional LSTM predictor with smaller receptive field is presented in this section. In the sequel, the total number of subcarriers covered by the receptive field of such a predictor is denoted by F_0, with $F_0 < F$, and the corresponding predictor is referred to as F_0-convolutional LSTM predictor.

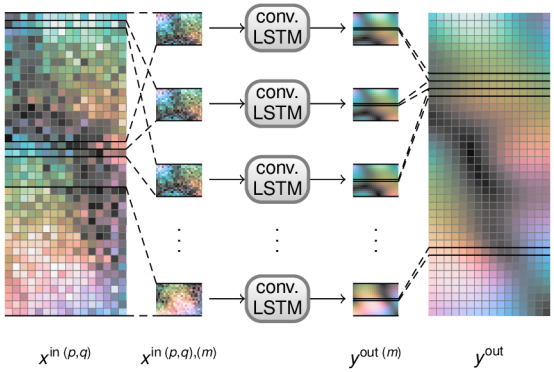

FIGURE 4. Operation of F_0-convolutional LSTM predictor.

The prediction scheme of the F_0-convolutional LSTM is illustrated in FIGURE 4 and described in the following. First, each of the original input tensors $x^{\text{in}\,(p,q)}$ in (1) is partitioned into F-many overlapping segments, $x^{\text{in}\,(p,q),(m)}$ for $m = 0, \ldots, F-1$, where the frequency axis f of each segment $x^{\text{in}\,(p,q),(m)} = \{x^{\text{in}\,(p,q),(m)}_{t,\lambda,f}\}_{t,\lambda,f}$ ranges from $m - (F_0 - 1)/2$ to $m + (F_0 - 1)/2$. (Here, for simplicity F_0 is assumed to be odd.) In the case where one of these bounds falls outside of the originally considered spectrum, the corresponding entries are filled with zeros. For each m, the tensors $x^{\text{in}\,(p,q),(m)}$ for all indices (p, q) are processed through the convolutional LSTM from Section II-C and the neural network output tensor is denoted by $y^{\text{out}\,(m)} = \{y^{\text{out}\,(m)}_{t,\lambda,f}\}_{t,\lambda,f}$. The final neural network output of the F_0-convolutional LSTM, $y^{\text{out}} = \{y^{\text{out}}_{t,\lambda,f}\}_{t,\lambda,f}$, defined in accordance with (2), is obtained by taking those entries of each segment $y^{\text{out}\,(m)}$ that are located on the central subcarrier m and concatenating them along the frequency axis over all m, i.e.,

$$y^{\text{out}}_{t,\lambda,f} := y^{\text{out}\,(f)}_{t,\lambda,f}$$

for all t, λ, and f (substituting $m = f$).

Note that, in the actual algorithm, for each m, the output layer of the convolutional LSTM only consists of connections that directly link to the values $y^{\text{out}\,(m)}_{t,\lambda,f}$ with $f = m$, as only the central subcarrier m of each segment $y^{\text{out}\,(m)}$ is relevant for the computation. This allows for a reduction of the number of operations required for the prediction when implementing this type of predictor.

Note that by transforming the original convolutional LSTM predictor into the F_0-convolutional LSTM predictor, the size of the receptive field in the frequency domain, F_0, is made a tunable parameter. Following the discussion in Section II-C.2, F_0 is set to 17 which leads to a predictor whose receptive field more closely matches that of the partially dilated CNN predictor. The receptive field of this F_0-convolutional LSTM predictor is illustrated in FIGURE 2 by the red area filled with horizontal lines.

E. COMBINED NEURAL NETWORK

Experiments in Section V show that among the preceding predictors, for long-term predictions (i.e. for r near R), the partially dilated CNN predictor is advantageous over both of the LSTM-based predictors, whereas for short-term predictions (i.e. for r near 1), the F_0-convolutional LSTM delivers the most accurate results. This suggests a possibility of overall performance improvement by seeking appropriate linear combinations of the outputs from the best predictors.

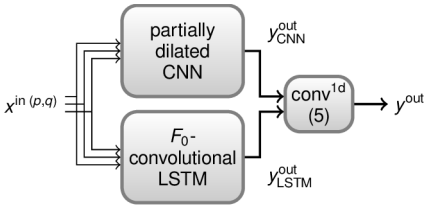

FIGURE 5. Layout of combined neural network predictor.

As illustrated in FIGURE 5, the combined neural network predictor consists of the partially dilated CNN predictor, the F_0-convolutional LSTM predictor, and an extra one-dimensional convolutional layer. For each time step t, the two-dimensional output tensors $\{y^{\text{out}}_{t,\lambda,f}\}_{\lambda,f}$ of the partially dilated CNN predictor and the F_0-convolutional LSTM predictor are concatenated along the channel axis λ and passed through the extra one-dimensional convolutional layer convolving along the frequency axis f, that is, in (4), let $d = 1$ and let $\star$ represent the f-axis, with $(P \cdot Q \cdot 2 \cdot R)$-many output channels and kernel size 5, using no activation function.

Since the combined neural network is based on two existing predictors, these two preceding predictors can be trained separately prior to being combined. For training the combined neural network, it then suffices to adjust only the parameters of the final convolutional layer, which is a much less time intensive task than simultaneously training all of the parameters and also allows for a more memory-intensive training algorithm (e.g. L-BFGS) to be used (cf. Section III).

III. TRAINING ALGORITHM

Recall that each of the neural networks introduced in Section II defines a predictor $\Gamma(\cdot, \boldsymbol{\xi})$ parametrised by a vector of neural network parameters $\boldsymbol{\xi} \in \mathbb{R}^{\Xi}$. The vector $\boldsymbol{\xi}$ consists of all the entries appearing in the convolutional kernels and bias vectors of the neural network. In supervised learning, the free parameters of a predictor are adjusted to all available labelled *training examples* (i.e. training inputs paired with training targets) by minimising the average distance between the predictions and the corresponding target values across all training inputs. Such distance is measured by the so-called *loss function*. The loss function used in this work is the mean squared error of the multi-step ahead predictions delivered by the predictor. Given time series H^{est} and H^{true} as introduced in the beginning of Section II, the mean squared error $\eta^{\Gamma}((H^{\mathrm{est}}, H^{\mathrm{true}}), \boldsymbol{\xi})$ of the predictor $\Gamma(\cdot, \boldsymbol{\xi})$ on these time series can be computed via

$$\eta^{\Gamma}((H^{\mathrm{est}}, H^{\mathrm{true}}), \boldsymbol{\xi}) := \operatorname*{mean}_{p,q,t,r,f} \left| \Gamma(\{H^{\mathrm{est}}_{t'}\}_{t' \leq t}, \boldsymbol{\xi})^{(p,q)}_{r}[f] - H^{\mathrm{true}\,(p,q)}_{t+r}[f] \right|^2 \quad (6)$$

where the indices p, q, r, and f range over all available transmitter antenna ports, receiver antennae, prediction time steps, and subcarriers, respectively. (Recall the data transformations between the complex-valued time series and the real-valued neural network input and output tensors in (1) and (2).) In (6), for the temporal index t, only indices that are at least as large as a specified initial period T_0 are used to form the mean, so that the predictor is given enough historical information to produce meaningful predictions. In the present case, a value of $T_0 = 255$ is chosen, which coincides with the size of the cumulative local receptive field of the partially dilated CNN predictor along the temporal axis. The total prediction span R is set to 10.

The parameters of the proposed neural networks are adjusted by means of refined versions of the *stochastic gradient descent* (SGD) algorithm. In SGD, the loss values $\eta^{\Gamma}((H^{\mathrm{est}}, H^{\mathrm{true}}), \boldsymbol{\xi})$ are averaged over a certain amount of training examples in each update step and improved by moving the parameter vector $\boldsymbol{\xi} \in \mathbb{R}^{\Xi}$ locally in direction of the negative gradient of the averaged loss. More specifically, the training dataset $\mathcal{X}$ containing pairs of training inputs and targets $(H^{\mathrm{est}}, H^{\mathrm{true}})$ is randomly partitioned into disjoint subsets $\{\mathcal{X}_i^{\mathrm{batch}}\}_{i \in I}$ called *mini-batches* that are passed successively through the neural network being trained. Each pass through the entire training dataset $\mathcal{X}$ is called an *epoch*. At the beginning of each epoch, a new random partition $\mathcal{X} = \dot{\bigcup}_{i \in I} \mathcal{X}_i^{\mathrm{batch}}$ into mini-batches is chosen. In order to include more variety in each of the mini-batches, each recording of the time series $(H^{\mathrm{est}}, H^{\mathrm{true}})$ in the training dataset $\mathcal{X}$ is subdivided into 8 overlapping segments. These segments overlap in such a manner that the available prediction for each time step of the form $t + r$ with $t \geq T_0$ and $r = 1, \ldots, R$ is included exactly once in an epoch in the calculation of the loss value. Now, beginning with a randomly initialised parameter vector $\boldsymbol{\xi} \in \mathbb{R}^{\Xi}$, the parameter update rule for each mini-batch reads as

$$\boldsymbol{\xi} \leftarrow \boldsymbol{\xi} - \gamma \cdot \operatorname*{mean}_{(H^{\mathrm{est}}, H^{\mathrm{true}}) \in \mathcal{X}_i^{\mathrm{batch}}} \nabla_{\boldsymbol{\xi}} \eta^{\Gamma}((H^{\mathrm{est}}, H^{\mathrm{true}}), \boldsymbol{\xi})$$

where $\gamma > 0$ is a scaling factor called *learning rate* and $\nabla_{\boldsymbol{\xi}}$ refers to the gradient operator with respect to the parameter vector $\boldsymbol{\xi}$. The gradient value $\nabla_{\boldsymbol{\xi}} \eta^{\Gamma}((H^{\mathrm{est}}, H^{\mathrm{true}}), \boldsymbol{\xi})$ is computed in an efficient manner via reverse-mode automatic differentiation, more specifically, via the *backpropagation* algorithm [14], which basically relies on the application of the chain rule of multivariable calculus.

All of the proposed neural networks except for the combined neural network are trained for 48 epochs using the Adam training algorithm by Kingma and Ba (2015) [34] which is a refined version of the SGD method. The learning rate γ is set to 0.01. The mini-batch size $\#\mathcal{X}_i^{\mathrm{batch}}$ is increased gradually during the training and is set to 2, 4, or 8 during the 16 epochs in the beginning, middle, or end of the training, respectively. The combined neural network is trained for 48 epochs using the quasi-Newton L-BFGS algorithm of Liu and Nocedal (1989) [35] which, being a second-order method, generally exhibits faster convergence at the cost of high memory requirements. As the training of the combined neural network only requires the adjustment of a relatively low number of parameters (final convolutional layer), the L-BFGS algorithm is feasible for this task.

All computations are performed using the PyTorch machine learning framework developed by Paszke *et al.* (2019) [36].

IV. EVALUATION DATASETS

For training and evaluating the predictive neural networks, simulated transmissions over dispersive fading channels with Rayleigh components, cf. Barsocchi (2006) [37] and Schulze (1989) [38], are performed according to the reference measurement procedures defined in the well-established 3GPP Long Term Evolution (LTE) standards TS 36.101 [39] and TS 36.211 [29]. Accordingly, the estimated and actual transfer functions $H^{\mathrm{est}}, H^{\mathrm{true}}$ over a certain period of time are recorded and collected into the training and test datasets. The simulation is implemented in the Matlab computing environment, partially making use of the LTE Toolbox [40]. The data generation scheme and the resolution of the time series $H^{\mathrm{est}}, H^{\mathrm{true}}$ used herein are summarised below and described in more detail in the supplementary material.

Throughout the simulation, OFDM symbols distributed across 600 subcarriers (i.e. 50 resource blocks) spanning

TABLE 1. Mean squared errors of proposed and reference predictors in *dB*.

		Prediction span (r)										
	Predictor	1	2	3	4	5	6	7	8	9	10	Total
A	part. dil. CNN	−16.93	−16.03	−14.92	−13.75	−12.60	−11.46	−10.38	−9.35	−8.38	−7.49	−11.10
	conv. LSTM	−17.07	−16.00	−14.74	−13.43	−12.14	−10.90	−9.72	−8.63	−7.63	−6.72	−10.48
	F_0-conv. LSTM	−17.27	−16.18	−14.94	−13.63	−12.33	−11.09	−9.92	−8.82	−7.82	−6.90	−10.67
	combined	−17.34	−16.49	−15.42	−14.28	−13.11	−11.99	−10.90	−9.85	−8.87	−7.94	−11.58
	trivial	−7.54	−5.92	−4.21	−2.65	−1.31	−0.16	0.82	1.65	2.35	2.94	−0.26
B	RNN, $F_0 = 1$	−13.07	−11.89	−10.64	−9.42	−8.27	−7.23	−6.28	−5.43	−4.68	−4.02	−7.18
	NARX, $F_0 = 1$	−11.43	−10.03	−8.69	−7.53	−6.45	−5.43	−4.48	−3.60	−2.81	−2.14	−5.34
	LSTM, $F_0 = 1$	−13.63	−12.53	−11.38	−10.24	−9.15	−8.13	−7.18	−6.32	−5.54	−4.84	−8.03
	KF, $F_0 = 1$	−12.90	−11.39	−9.88	−8.43	−7.10	−5.89	−4.82	−3.87	−3.05	−2.34	−5.78
C	time domain AR/KF	−14.41	−12.44	−10.38	−8.22	−5.30	0.60	9.90	20.30	30.90	41.54	31.93
D	LSTM, $F_0 = 200$	−5.71	−4.79	−3.72	−2.67	−1.72	−0.91	−0.24	0.29	0.69	0.97	−1.26
	LSTM, $F_0 = 17$	−17.28	−16.26	−15.01	−13.69	−12.37	−11.10	−9.90	−8.78	−7.74	−6.79	−10.63

a bandwidth of 10 MHz are transmitted in a SISO or MIMO transmission mode, following the LTE standards [29], [39]. Normal cyclic prefix is used, which corresponds to a symbol rate of 14 OFDM symbols per millisecond (i.e. 7 OFDM symbols per slot). For each antenna configuration, the channel transmission is simulated in a first step independently for each of the individual transmitter-receiver antenna pairs, following the stochastic model for dispersive fading with Rayleigh components [37], [38] approximated by the generalised method of exact Doppler spread by Pätzold *et al.* (2009) [41], with amplitudes, path delays, and maximum Doppler frequencies chosen according to [39, B.2.1, B.2.2]. Subsequently, the transmitter-receiver antenna pairs are mixed according to one of the three reference examples of MIMO correlation matrices given in [39, B.2.3]. On top of that, for a given target signal to noise ratio (SNR), Gaussian white noise with corresponding variance is added to each transmitted signal. The training algorithm presented in this work uses the actual channel transfer function H^{true} as ground truth. In the simulation, H^{true} is obtained by transmitting an impulse train from each of the transmitter antennas in separate copies of the same simulation environment and recording the received noise-free signals. For training a predictor using data originating from a real-world system, H^{true} can be replaced by H^{est} which slightly decreases the prediction accuracy.

For each seed of the simulation, a transmission period of 400 ms (i.e. 40 frames) in total is recorded and cell-specific reference signals according to [29] are used for the channel estimation. Due to the nature of the prescribed reference signals, the estimated channel transfer function is evaluated slot-wise and every 3 subcarriers. This results in 800 time steps and $F = 200$ subcarriers in total for each recording of the time series H^{est} and H^{true}. With this temporal resolution, the total prediction span of $R = 10$ time steps corresponds to a time period of 5 ms (i.e. one half-frame).

Overall, for each simulation setting specified in terms of SISO or MIMO setting, MIMO correlation, delay profile, maximum Doppler frequency, and target SNR, 12 seeds in total are used to generate the random transmission channel along with the recordings (H^{est}, H^{true}) for training one predictive neural network. Here, the cell IDs used for determining the cell-specific reference signals are chosen such that each equivalent class of cell ID corresponding to the same set of resource elements assigned to the reference signals appears twice. Accordingly, another 6 seeds and an extra single seed are used for generating the test and validation data, respectively.

V. RESULTS

In this section, the predictive neural networks presented in Section II are evaluated on the datasets introduced in Section IV and compared to some of the common reference predictors. In the methodical comparative studies in Sections V-A and V-B, a default channel configuration, with 4 × 2 antenna configuration, EVA70 propagation conditions (cf. [39, B.2.1, 2.2]), medium MIMO correlation (cf. [39, B.2.3]), and an SNR of 9 dB, is used for both training and evaluating the predictors. In the cross-scenario tests in Section V-C, a fixed class of predictors corresponding to different antenna configurations are trained and tested across a variety of channel configurations.

In accordance with the loss function defined in (6) which is optimised through training, the performance of each considered predictor is measured by the mean squared error (MSE) of the prediction. The MSE is reported in decibels (*dB*); as the time-varying transfer function H^{true} is normalised to an expected power of 1, this value coincides with the relative squared prediction error.

A. PERFORMANCE COMPARISON

The subsequent comparative evaluation includes a performance analysis of the proposed convolutional-type neural networks and comparisons with three classes of reference methods: a straightforward adaption of existing single-subcarrier predictors, the OFDM-specific time-domain approach proposed in [6], and an alternative (non-convolutional) solution to the multi-subcarrier problem. A brief description of the reference methods is provided in Appendix A. The results of the comparative study are presented in TABLE 1, illustrated in FIGURE 6, and interpreted in the following.

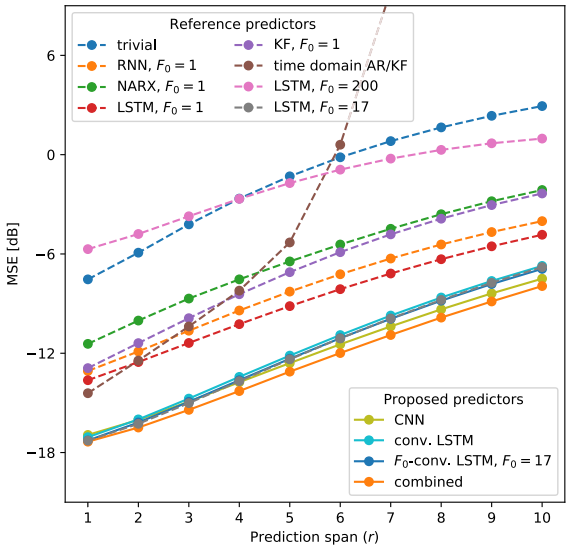

FIGURE 6. **Performance comparison of proposed and reference predictors.**

1) PERFORMANCE OF PROPOSED NEURAL NETWORKS

The performance evaluation of the proposed convolutional-type neural networks is presented in Part A of TABLE 1. Over the total prediction span of $R = 10$ time steps, the partially dilated CNN predictor is superior to both the regular and the F_0-convolutional LSTM predictors in the later steps $r \gg 1$. Vice versa, both of the convolutional LSTM predictors perform better than the partially dilated CNN predictor in the earlier steps $r \ll R$. Among the two LSTM-based predictors, the F_0-convolutional LSTM outperforms the convolutional LSTM in each prediction step r by around 0.2 dB (5%) on average. Overall, the combined neural network exhibits the highest prediction accuracy among all predictors across all prediction steps.

Note that for the first 7 prediction steps, all proposed neural networks achieve prediction errors that are below the noise level of the input signal, i.e., MSE $< -\text{SNR} = -9$ dB, which displays the intrinsic denoising capability of these predictors.

2) COMPARISON WITH PREVIOUS SINGLE-SUBCARRIER PREDICTORS

As stated in the introduction Section I-A, previous contributions to fading channel forecasting mainly focused on single-subcarrier settings and employed common predictors such as the Kalman filter (KF) for autoregressive (AR) channel modelling, the standard recurrent neural network (RNN), the nonlinear autoregressive exogenous model (NARX) RNN, and the standard LSTM. In the present evaluation, each of these single-subcarrier predictors is straightforwardly adapted to the present OFDM setting by treating each subcarrier $f = 0, \ldots, F-1$ separately and passing each single-subcarrier observation series of the form $\{H_t^{\text{est}}[f]\}_t$ through the same predictor with $F_0 = 1$ (cf. Appendix A-A, A-C, A-D, A-E for more details). From Part B of TABLE 1, with the standard LSTM with $F_0 = 1$ performing the best among all considered single-subcarrier predictors, all of these predictors deliver significantly lower prediction accuracy than any of the proposed convolutional-type neural networks over the entire prediction span.

Evidently, employing a joint single-subcarrier predictor shared across the entire bandwidth in the spectrum is not an appropriate approach for treating OFDM transmission systems. As more advanced alternatives, the subsequent reference predictors either operate in the time domain and employ separate predictive models associated with different delay times in the channel impulse response or incorporate more subcarriers into the receptive field of the predictor (i.e. $F_0 > 1$).

3) COMPARISON WITH PREVIOUS TIME-DOMAIN APPROACH

In [6], a method that uses AR channel models in the time domain was designed specifically for fading channel forecasting in OFDM transmission systems. The method is based on inverse Fourier transforming the observed channel transfer functions into observation series in the time domain and employing separate AR models for different delay times in the time series of channel impulse responses (cf. Appendix A-B for more details). The key observation motivating such a method is that, by passing to a time domain representation of the available channel state information, the interdependencies between different subcarriers in the frequency domain (correlation bandwidth) disappear. As presented in [6], the method does not take into account the presence of noise in the observations and uses a technique for fitting the AR models which is not directly applicable in the present case. As remarked in Section I-A, each ARIMA model with Gaussian disturbance has an equivalent linear state-space model and thus, a natural extension of the method is the application of a Kalman filtering scheme using the prediction error method for fitting the parameters.

From Part C of TABLE 1, for one-step ahead prediction $r = 1$, the time-domain AR/KF method of [6] performs better than any of the single-subcarrier reference predictors (compare Part B of TABLE 1), yet worse than any of the proposed predictors (Part A of TABLE 1); for later prediction steps $r \gg 1$, however, the method appears to be unstable and the MSE explodes when increasing the prediction span r. A potential cause of the unstable behaviour in the later steps is the unequal distribution of power over the impulse response, which entails the presence of large values at delay times near zero. Considering the fact that power is distributed more uniformly across the spectrum, such problem does not affect predictors that work directly in the frequency domain as proposed in this work.

4) COMPARISON WITH ALTERNATIVE MULTI-SUBCARRIER APPROACH

In order to achieve more comparable reference performance and as a representative natural extension of the existing

single-subcarrier methods, the best among them, the standard LSTM predictor (recall V-A.2), is now endowed with higher input dimension which is associated with larger receptive fields in the frequency domain, $F_0 = F = 200$ or $F_0 = 17$. Accordingly, the resulting fully-connected LSTM is referred to as standard LSTM predictor with $F_0 = 200$ or $F_0 = 17$ (cf. Appendix A-C for more details). The evaluation of this type of predictor is presented in Part D of TABLE 1.

Compared to the single-subcarrier predictors, the standard LSTM predictor with $F_0 = F = 200$ performs even worse and is heavily overfitted, which results from the huge number of parameters required in the fully-connected layers for treating the entire bandwidth $F = 200$ as a whole. This particularly contrasts with the proposed convolutional LSTM predictor that uses the same size of receptive field $F_0 = F$ in the frequency domain but requires significantly fewer parameters by employing shared weights in the frequency domain.

Among all reference predictors, the standard LSTM predictor with $F_0 = 17$ exhibits the best performance which is nearly as good as the overall performance of the proposed F_0-convolutional LSTM predictor with the same local receptive field. However, the high performance of such a fully-connected LSTM is achieved at the price of much higher computational effort compared to all of the proposed convolutional-type neural networks (see Section V-B for more details).

Remark: It is worth noting that as an advantage over the traditional iterative procedure for multi-step ahead predictions as used in the KF and NARX, the predictive neural networks proposed in this work are able to deliver all multi-step ahead predictions over the entire horizon R simultaneously and avoid inherited errors. In fact, by employing separate channels associated with different prediction time steps $r = 1, \ldots, R$ in the output layer of the predictive neural network, the prediction errors over the entire horizon R are included simultaneously in the loss function and minimised jointly during the training. Note that such network layout is also implemented in all of the reference predictors except for KF and NARX. Therefore, the benefit of the proposed treatment of multi-step ahead predictions can be observed when comparing all reference predictors with $F_0 = 1$ (as it would be fair to compare predictors with the same level of performance): In FIGURE 6, the plots of MSEs over $r = 1, \ldots, R$ for the KF and NARX with $F_0 = 1$ exhibit higher slopes than those for the RNN and LSTM with $F_0 = 1$.

B. COMPUTATIONAL COMPLEXITY

TABLE 2 displays information regarding the computational complexity of the proposed predictors and the reference predictors considered in Section V-A. The number of multiplications is given per time step and per subcarrier and relates to the naïve implementation without any fast convolution or fast matrix multiplication algorithms which could be used in an application requiring higher efficiency. For the time domain KF approach, the number of multiplications (marked by *) does not include the multiplications performed in the two required Fourier transforms, whose number depends approximately logarithmically on the total number of subcarriers F and is low compared to the effort of the remaining parts of the algorithm. The number of additions required by the predictors is very similar to the number of required multiplications and therefore not included in the table. All activation functions in the considered neural networks are either hyperbolic tangent or sigmoid activations which require almost the same computation time, ensuring the comparability of the results.

TABLE 2. Computational complexity of proposed and reference predictors.

	Predictor	Parameters	Multiplications	Activations
A	part. dil. CNN	138976	138048	384
	conv. LSTM	104608	103968	480
	F_0-conv. LSTM	104608	538656	8160
	combined	499744	932704	8544
B	RNN, $F_0 = 1$	17056	16704	96
	NARX, $F_0 = 1$	752	7040	160
	LSTM, $F_0 = 1$	21664	21024	480
	KF, $F_0 = 1$	3329	499200	
C	time domain AR/KF	665800	499200*	
D	LSTM, $F_0 = 200$	235603200	1177696	160
	LSTM, $F_0 = 17$	1582240	1573920	8160

According to Part A of TABLE 2, among the proposed predictors, the partially dilated CNN predictor and the convolutional LSTM predictor are computationally most efficient. Note that even though the convolutional LSTM predictor requires the fewest number of parameters and multiplications, its sequential nature limits the degree of parallelisation of computation that can be performed during training, which makes the training process more time consuming than that of the feedforward CNN predictor. Compared to the regular convolutional LSTM predictor, the F_0-convolutional LSTM predictor uses a computation scheme that maintains separated states for each objective subcarrier, which accounts for the higher computational effort required during inference. This also applies to the combined neural network predictor, given its nature of combining both the performance-wise advantages and the computational disadvantages of the preceding predictors. Overall, the numbers indicate that running the proposed predictors in real-time is indeed possible on modern hardware. More precisely, an entry-level graphics processing unit (GPU) (as of September 2020) such as the Nvidia GeForce GTX 1650 with roughly 2.5 T floating point operations per second (FLOPS) can perform the computations required for the combined predictor at about 20% load, even without further optimisation of the operations involved and more energy efficient implementations on dedicated hardware, possibly using fixed-point arithmetic, are certainly possible.

Considering Part B of TABLE 2, among the single-subcarrier reference predictors considered in Section V-A.2, the methods employing RNN, NARX, and LSTM with $F_0 = 1$ require much less computational effort compared

to the proposed predictors. Such computational efficiency, however, relies on the single subcarrier modelling which is, according to the performance evaluation in Part B of TABLE 1, not complex enough for treating OFDM transmission systems. The computational complexity of the KF-based single-subcarrier predictor is close to that of the proposed F_0-convolutional LSTM.

According to Part C of TABLE 2, the OFDM-specific time-domain method considered in Section V-A.3 has the same complexity as the single-subcarrier KF (not including the computational effort of the Fourier transforms, which is minor) and is therefore comparable to the F_0-convolutional LSTM in respect of computational complexity. However, as displayed in Part C of TABLE 1, the method only achieves high accuracy across low prediction spans which does not justify the relatively high computational complexity.

According to Part D of TABLE 2, among the multi-subcarrier reference predictors considered in Section V-A.4, the standard LSTM predictor with $F_0 = 200$ requires an extremely large number of parameters, which translates to a very high memory requirement, resulting from the fact that such a fully-connected network does not efficiently share parameters across subcarriers. Finally, the performance-wise most competitive reference predictor, the standard LSTM predictor with $F_0 = 17$ (cf. Part D of TABLE 1), requires more computational resources than any of the proposed predictors. In particular, it uses more than 15 times as many parameters and 3 times as many multiplications as the F_0-convolutional LSTM with the same size of receptive field in the frequency domain.

C. CROSS-SCENARIO PERFORMANCE

While in Sections V-A and V-B, different prediction schemes are evaluated on the same dataset related to a default channel configuration, in this section, a representative class of predictors are trained on and tested across different datasets associated with different channel configurations. Conducting such cross-scenario tests aims to verify the general applicability of the proposed neural networks and assess the impact of the channel parameters on the predictor performance. Considering the computational effort required during the extensive tests, in particular during the training phases, the partially dilated CNN predictors, which are the fastest to train, are used as the representative predictors. In accordance with [39, B.2.1-2.3] (recall the data generation scheme introduced in Section IV), the simulation settings considered herein include the following channel parameters: antenna configuration, delay profile, maximum Doppler frequency, MIMO correlation, and SNR, with available values summarised in TABLE 3. Each time varying one of these channel parameters, the others are set to their default values as used in Sections V-A and V-B (highlighted in TABLE 3).

Generally, the cross-scenario tests verify that the proposed predictors are compatible with all of the considered simulation settings. As a common result, the predictor performance increases with rising similarity between the training and test

TABLE 3. Simulation settings for cross-scenario tests with default configuration highlighted in bold.

Antenna configuration	MIMO correlation	Propagation conditions / Doppler frequency	SNR
1×1	Low	EPA5	21 dB
1×2	**Medium**	EVA5	18 dB
2×2	High	ETU30	15 dB
$\mathbf{4 \times 2}$		**EVA70**	12 dB
4×4		ETU70	**9 dB**
		ETU300	6 dB
			3 dB
			0 dB
			−3 dB

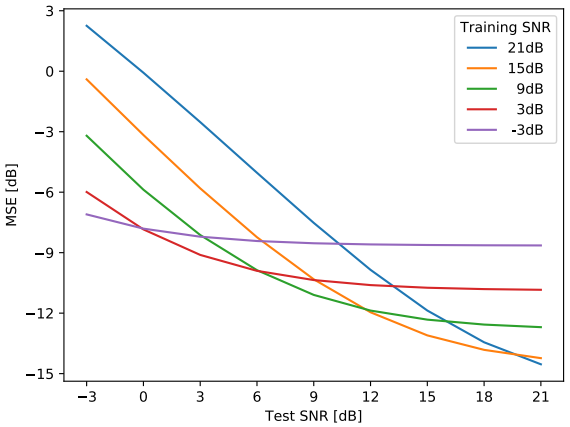

FIGURE 7. Performance of predictors trained on and tested across datasets with different SNRs.

channel configurations. Such behaviour is clearly observed in the evaluation with varying SNRs, as illustrated in FIGURE 7, where the different color lines represent predictors that are trained on datasets associated with different SNRs. Similar observations apply to the tests with varying MIMO correlations, with the main difference being that the precision of predictors trained on lower MIMO correlation levels does not increase when tested on a dataset with higher MIMO correlation. Concerning the propagation conditions, while the prediction accuracy varies substantially across the three different delay profiles, the choice of delay profile for the training data does not seem to have a large impact on the resulting predictor. More specifically, the predictors trained on the EPA5 and EVA5 datasets could both be used almost equally well in both of these propagation conditions and the same applies to the EVA70 and ETU70 propagation conditions.

Finally, the results of the tests across different antenna configurations are displayed in TABLE 4. Note that the prediction error decreases by 0.7 dB for the higher antenna count configurations, which indicates that the predictor is able to utilise the correlation of MIMO channels.

VI. DISCUSSION

In order to assess and compare the capability of different predictors, in this work, each of the proposed predictive neural networks is treated as a time-invariant model and trained

TABLE 4. Mean squared errors of convolutional predictor across different antenna configurations.

Antenna configuration	Transmitter antenna port 0	1	2	3	Total
1×1	-10.27				-10.27
1×2	-10.46				-10.46
2×2	-10.86	-10.63			-10.75
4×2	-11.48	-11.77	-11.16	-10.15	-11.10
4×4	-11.01	-11.85	-11.05	-10.23	-11.00

offline on a dataset associated with certain propagation conditions, which implicitly assumes a weak form of homogeneity in time of the observation and target time series $H^{\text{est}}, H^{\text{true}}$. According to the results of the cross-scenario tests from Section V-C, slight deviations in the channel parameters do not pose a significant problem to the predictors. Since the channel parameters usually do not experience abrupt changes, multiple predictors can be used to bridge the transition time between different settings. Thus, one possible way to adapt the proposed approach to a setting with time-variant channel conditions would be to train preliminarily a set of predictive neural networks, each associated with a relevant class of fading channels (e.g. with graded maximum Doppler frequencies, MIMO correlations, and SNRs within a certain range), and estimate or categorise in the test phase the actual ongoing channel parameters at regular intervals by employing either well-established classical methods (e.g. Hansen *et al.* (1999) [42]) or machine learning-based clustering or classification schemes (e.g. Ahrens *et al.* (2019) [43] or Yuan *et al.* (2020) [25]), so that the forecasts can be delivered by combining outputs from predictors that are trained for the most closely matching channel classes. This idea resembles that of [25] where a CNN is used as a channel classifier that assigns a pre-trained predictor with matching channel characteristics to the observation series.

VII. CONCLUSION

In this article, modern machine-learning techniques are adopted for the issue of multi-step fading channel forecasting in OFDM transmission systems. By exploiting the translation-invariance property of the convolution operation, a variety of predictive neural networks that incorporate convolutional layers for processing the channel state information in the frequency domain are proposed for the task, which provides an effective approach to the more complex multi-subcarrier setting. The comparative evaluation of both the traditional stochastic approach employing the Kalman filtering scheme and standard machine learning methods employing well-established predictive neural networks confirms the advantage of the proposed approach.

APPENDIX A

This section provides a brief description of the reference predictors considered in the performance analysis in Section V-A. Here, let F_0 with $F_0 \in \{1, 17, F = 200\}$ again denote the size of the receptive field of the considered predictor in the frequency domain. In case of $F_0 \in \{1, F\}$, the time series $H^{\text{est}}, H^{\text{true}}$ are considered as F/F_0-many $\mathbb{R}^{P\cdot Q\cdot 2\cdot F_0}$-valued time series. In case of $F_0 = 17$, analogously to the input data transformation used in the F_0-convolutional LSTM predictor as introduced in Section II-D, the original observation series H^{est} is transformed into F-many $\mathbb{R}^{P\cdot Q\cdot 2\cdot F_0}$-valued overlapping time series, $\{H^{\text{est}(f)}\}_{f<F}$, each covering F_0-many subcarriers and being processed through the considered predictor for forecasting H^{true} on a single subcarrier f.

A. KALMAN FILTERING FOR AR CHANNEL MODELLING

When considering relatively short time spans (hundreds of milliseconds) with no abrupt changes in the propagation conditions, the time series of the perfect channel transfer function can be assumed to follow a multivariate AR model (cf. e.g. [4]). For $F_0 = 1$, the corresponding observation model can be formulated as

$$\begin{aligned} H_t^{\text{true}}[f] &= \sum_{k=1}^{\mathcal{P}} \phi_k H_{t-k}^{\text{true}}[f] + v_t^f \\ H_t^{\text{est}}[f] &= H_t^{\text{true}}[f] + w_t^f \end{aligned} \tag{7}$$

for all t and f, where $\phi_k \in \mathbb{R}^{P\cdot Q\cdot 2\times P\cdot Q\cdot 2}$ for each k and $\{v_t\}_t, \{w_t\}_t$ are $\mathbb{R}^{P\cdot Q\cdot 2\times F}$-valued independent white noise processes, each consisting of independent and identically distributed (i.i.d.) zero-mean Gaussian random tensors with i.i.d. entries. Transforming (7) into an equivalent linear state-space model analogously to [2, Ch. 4.3] and estimating the parameters by means of the prediction error method (cf. e.g. [3, Ch. 7.4]), the Kalman filtering scheme (cf. e.g. [2, Ch. 4.5] or [3, Ch. 5.5]) delivers the one-step ahead prediction of H^{true} which is for each time step t the conditional expectation of H_{t+1}^{true} given observations H^{est} up to time t. The multi-step ahead predictions are computed iteratively. For the default antenna configuration $P \times Q = 4 \times 2$, considering the computational effort and memory intensity, the order $\mathcal{P}$ is set to 13.

B. KALMAN FILTERING FOR AR CHANNEL MODELLING IN TIME DOMAIN

The method presented in [6] proceeds by first applying an inverse Fourier transform to the channel transfer functions $H^{\text{est}}, H^{\text{true}}$, resulting in the channel impulse responses $h^{\text{est}}, h^{\text{true}}$, i.e.,

$$h_t[\Delta\tau] = \frac{1}{F}\sum_{f=0}^{F-1} H_t[f]\exp\left(2\pi\,\Delta\tau\frac{f}{F}\right) \tag{8}$$

for all t, $\Delta\tau = 0, \ldots, F-1$, and for $h = h^{\text{true}}, h^{\text{est}}$ and $H = H^{\text{true}}, H^{\text{est}}$, respectively. For each delay time $\Delta\tau = 0, \ldots, F-1$, a separate AR model in the form of (7) is employed where $H^{\text{est}}, H^{\text{true}}$, and f are replaced by $h^{\text{est}}, h^{\text{true}}$, and $\Delta\tau$, respectively. Then, the predictions are computed in the time domain and transferred back into the frequency domain via a Fourier transformation.

C. STANDARD LSTMs

Note that in a fully-connected neural network, each layer connection can be expressed in terms of a one-dimensional convolutional layer connection with kernel size 1 (recall (4)) by relating the dimension of the input or output vector of the former to the number of the input or output channels of the latter, respectively, and adding an extra axis of size 1 to the original input vector. Using the same terminology as in Section II, for $F_0 \in \{1, 17\}$ or $F_0 = F = 200$, the corresponding comparative predictor employing a standard (i.e.fully-connected) LSTM [13] is endowed with $6 \cdot P \cdot Q \cdot 2 \cdot F_0 \times 1$- or $2 \cdot P \cdot Q \cdot 2 \cdot F \times 1$-dimensional hidden states and designed with architecture analogous to that introduced in Section II-D or Section II-C, respectively. Note that in these reference predictors, the treatment of MIMO channels and the computation of multi-step ahead predictions are in accordance with those implemented in the proposed neural networks, which again performs better than comparable alternatives.

D. STANDARD RNN

Standard (i.e. fully-connected) RNNs can be seen as fully-connected LSTMs without gates and cell values [12]. The comparative predictor employing a standard RRN is designed and trained in the same manner as the standard LSTMs introduced above.

E. NONLINEAR AUTOREGRESSIVE EXOGENOUS MODEL (NARX) RNN

In [25], Yuan *et al.* present a channel forecasting scheme consisting of a channel classifier followed by a pre-trained predictor. While the article mainly focuses on the channel classification part (using a CNN) along with the supporting channel estimation scheme, the included predictors are interesting in their own right and can be evaluated as single-subcarrier predictors on the dataset used in this article. The two proposed predictors in [25] employ an AR model on the one hand and a modified RNN forming a NARX model as introduced in Lin *et al.* (1996) [44] on the other hand. While the AR approach using Kalman filtering scheme has already been considered above, the NARX RNN is adapted to the present setting as described below. The NARX RNN is a single subcarrier predictor whose hidden state dimension is the same as the input dimension. The hidden state is a nonlinear function of the current input and the last $\mathcal{P}$-many hidden states. The architecture allows for a particularly low parameter count and relatively low computational effort. For comparison, the delay time $\mathcal{P}$ is set to 13 which matches the hyperparameter choice for the AR approach above and is experimentally confirmed to be a reasonable choice. In the present setting, the treatment of MIMO channels is performed in accordance with other neural networks considered in this article, i.e., by first performing forecasts for each antenna pair separately and then mixing these forecasts in a final output layer. As required for the NARX predictors, the multi-step ahead predictions are computed iteratively.

Remark: Note that in the present setting, in particular in the comparative study in Sections V-A, V-B, the proposed and reference predictors are trained and evaluated on a dataset associated with a pre-determined channel configuration. In particular, fixed channel conditions are assumed throughout the evaluation, so that no channel classifier as proposed in [25] is required prior to applying the actual predictor. Since the channel classifier is incorporated as a core component into the entire prediction scheme in [25], the comparative study in Sections V-A, V-B merely provides the net predictor performance, but not an assessment of the overall approach proposed in [25] compared to the approach proposed in this work. In Section VI, a possibility of combining a channel classifier such as that proposed in [25] with the more powerful predictive neural networks proposed herein is discussed for the more complex case with time-varying channel conditions.

REFERENCES

[1] J. Durbin and S. J. Koopman, *Time Series Analysis by State Space Methods* (Oxford Statistical Science Series), vol. 38, 2nd ed. Oxford, U.K.: Oxford Univ. Press, 2012.

[2] R. J. Elliott, L. Aggoun, and J. B. Moore, *Hidden Markov Models: Estimation Control* (Stochastic Modelling and Applied Probability), vol. 29, 1st ed. New York, NY, USA: Springer-Verlag, 1995.

[3] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA, USA: Holden-Day, 1970.

[4] A. Duel-Hallen, "Fading channel prediction for mobile radio adaptive transmission systems," *Proc. IEEE*, vol. 95, no. 12, pp. 2299–2313, Dec. 2007.

[5] A. Duel-Hallen, S. Hu, and H. Hallen, "Long-range prediction of fading signals," *IEEE Signal Process. Mag.*, vol. 17, no. 3, pp. 62–75, May 2000.

[6] D. Schafhuber and G. Matz, "MMSE and adaptive prediction of time-varying channels for OFDM systems," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 593–602, Mar. 2005.

[7] P. Sharma and K. Chandra, "Prediction of state transitions in Rayleigh fading channels," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, pp. 416–425, Mar. 2007.

[8] M. D. Zoltowski, M. Haardt, and C. P. Mathews, "Closed-form 2-D angle estimation with rectangular arrays in element space or beamspace via unitary ESPRIT," *IEEE Trans. Signal Process.*, vol. 44, no. 2, pp. 316–328, Feb. 1996.

[9] W. C. Jakes, Ed., *Microwave Mobile Communications*. New York, NY, USA: Wiley, 1974.

[10] S. Semmelrodt and R. Kattenbach, "Performance analysis and comparison of different fading forecast schemes for flat fading radio channels," Eur. Cooperation Field Sci. Tech. Res., Barcelona, Spain, Tech. Rep. COST 273 TD(03)045, 2003.

[11] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk, *A Distribution-Free Theory Nonparametric Regression* (Springer Series in Statistics). 1st ed. New York, NY, USA: Springer-Verlag, 2002.

[12] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

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

[15] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*. [Online]. Available: http://arxiv.org/abs/1609.03499

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

[17] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Trans. Cognit. Commun. Netw.*, vol. 4, no. 4, pp. 648–664, Dec. 2018.
[18] F. A. Aoudia and J. Hoydis, "End-to-End learning of communications systems without a channel model," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, USA, Oct. 2018, pp. 298–303.
[19] A. Felix, S. Cammerer, S. Dorner, J. Hoydis, and S. Ten Brink, "OFDM-autoencoder for end-to-end learning of communications systems," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Kalamata, Greece, Jun. 2018, pp. 1–5.
[20] H. Ye, G. Y. Li, B.-H.-F. Juang, and K. Sivanesan, "Channel agnostic End-to-End learning based communication systems with conditional GAN," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Abu Dhabi, United Arab Emirates, Dec. 2018, pp. 1–5.
[21] T. Ding and A. Hirose, "Fading channel prediction based on self-optimizing neural networks," in *Neural Information Processing* (Lecture Notes in Computer Science), vol. 8834, C. K. Loo, K. S. Yap, K. W. Wong, A. Teoh, and K. Huang, Eds. Cham, Switzerland: Springer, 2014, pp. 175–182.
[22] R.-F. Liao, H. Wen, J. Wu, H. Song, F. Pan, and L. Dong, "The Rayleigh fading channel prediction via deep learning," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–11, Jul. 2018.
[23] W. Jiang and H. D. Schotten, "Neural network-based fading channel prediction: A comprehensive overview," *IEEE Access*, vol. 7, pp. 118112–118124, 2019.
[24] W. Jiang and H. D. Schotten, "Deep learning for fading channel prediction," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 320–332, 2020.
[25] J. Yuan, H. Q. Ngo, and M. Matthaiou, "Machine learning-based channel prediction in massive MIMO with channel aging," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 2960–2973, May 2020.
[26] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, San Juan, Puerto Rico, 2016, pp. 1–14. [Online]. Available: https://dblp.org/db/conf/iclr/iclr2016.html
[27] J. Ahrens, L. Ahrens, and H. D. Schotten, "A machine learning method for prediction of multipath channels," *ZTE Commun.*, vol. 17, no. 4, pp. 12–18, 2019.
[28] A. Peled and A. Ruiz, "Frequency domain data transmission using reduced computational complexity algorithms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Denver, CO, USA, Apr. 1980, pp. 964–967.
[29] *Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation*, 3rd Gen. Partnership Proj.; Tech. Spec. Group Radio Access Network, Tech. Std. V9.1.0, 3GPP, document TS 36.211, Mar. 2010.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[31] A. Graves, "Generating sequences with recurrent neural networks," 2013, *arXiv:1308.0850*. [Online]. Available: http://arxiv.org/abs/1308.0850
[32] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. 29th Conf. Neural Inf. Process. Syst., NIPS 2015*, Montreal, QC, Canada, 2015, pp. 802–810.
[33] S. Zhang, Y. Wu, T. Che, Z. Lin, R. Memisevic, R. R. Salakhutdinov, and Y. Bengio, "Architectural complexity measures of recurrent neural networks," in *Proc. 30th Conf. Neural Inf. Process. Syst., (NIPS)*, Barcelona, Spain, 2016, pp. 1822–1830.
[34] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, 2015, pp. 1–15. [Online]. Available: https://dblp.org/db/conf/iclr/iclr2015.html
[35] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, nos. 1–3, pp. 503–528, Aug. 1989.
[36] A. Paszke, S. Gross, F. Massa, and A. Lerer, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. 33rd Conf. Neural Inf. Process. Syst. (NeurIPS)*, Vancouver, BC, Canada, 2019, pp. 8024–8035.
[37] P. Barsocchi, "Channel models for terrestrial wireless communications: A survey," CNR-ISTI, Pisa, Italy, Tech. Rep. ercim.cnr.isti//2006-TR-16, 2006.
[38] H. Schulze, "Stochastische modelle und digitale simulation von Mobilfunkkanälen," in *Kleinheubacher Berichte*, vol. 32. Darmstadt, Germany: Deutsche Bundespost Telekom, Fernmeldetechnisches Zentralamt, 1989, pp. 473–483.
[39] *Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Transmission and Reception*, 3rd Gen. Partnership Proj.; Tech. Spec. Group Radio Access Network, Tech. Std. V9.25.0, 3GPP, document TS 36.101, Sep. 2017.
[40] MathWorks, *LTE Toolbox: User's Guide*. Natick, MA, USA: MathWorks, 2019.
[41] M. Patzold, C.-X. Wang, and B. Hogstad, "Two new sum-of-sinusoids-based methods for the efficient generation of multiple uncorrelated Rayleigh fading waveforms," *IEEE Trans. Wireless Commun.*, vol. 8, no. 6, pp. 3122–3131, Jun. 2009.
[42] H. Hansen, S. Affes, and P. Mermelstein, "A Rayleigh Doppler frequency estimator derived from maximum likelihood theory," in *Proc. 2nd IEEE Workshop Signal Process. Adv. Wireless Commun.*, Annapolis, MD, USA, May 1999, pp. 382–386.
[43] L. Ahrens, J. Ahrens, and H. D. Schotten, "A machine-learning phase classification scheme for anomaly detection in signals with periodic characteristics," *EURASIP J. Adv. Signal Process.*, vol. 2019, no. 1, p. 27, Dec. 2019.
[44] T. Lin, B. G. Horne, P. Tino, and C. L. Giles, "Learning long-term dependencies in NARX recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1329–1338, Nov. 1996.

LIA AHRENS received the Dipl.-Math. degree in mathematics from the Technical University of Kaiserslautern, in 2011, and the Dr. rer. nat. degree in mathematics from Kiel University (CAU), Germany, in 2015, specializing in stochastics and financial mathematics. She is currently working as a Senior Researcher with the Department for Intelligent Networks, German Research Center for Artificial Intelligence, under the direction of Prof. Hans D. Schotten. Her research interests include stochastic calculus, stochastic filtering, and machine learning for time series analysis.

JULIAN AHRENS received the master's degree in mathematics from Kiel University (CAU), Germany, in 2016, specializing in non-commutative harmonic analysis. He is currently working as a Researcher with the Department for Intelligent Networks, German Research Center for Artificial Intelligence, under the direction of Prof. Hans D. Schotten. His research interests include high-performance computing, machine learning, digital signal processing, and harmonic and functional analysis.

HANS DIETER SCHOTTEN (Member, IEEE) received the Dipl.-Ing. and Dr.-Ing. degrees in electrical engineering from the RWTH Aachen University of Technology, Germany, in 1990 and 1997, respectively.

He is currently a Full Professor and the Director of the Institute for Wireless Communications and Navigation, Technical University of Kaiserslautern, Germany. He is also the Scientific Director of the German Research Center for Artificial Intelligence (DFKI) and the Head of the Department for Intelligent Networks. Since 2018, he has been the Chairman of the German Society for Information Technology and a member of the Supervisory Board of the VDE.