Segmented Spline Curve Neural Network for Low Latency Digital Predistortion of RF Power Amplifiers

Andrius Vaicaitis, Student Member, IEEE, and John Dooley, Member, IEEE

Abstract—At present, we are the dawn of a new era for wireless communication systems. The new dynamic approach to the operation of cellular networks requires an extension of the essential input–output hardware mechanisms, to include intelligence. Traditional neural network structures can be used to embed artificial intelligence into cellular network base stations; however, standard network structures are not an optimal solution for these use cases. In this article, we present a neural network structure, which is specifically designed to provide a more accurate and computationally efficient solution compared with the previous neural network solutions for predistortion of RF power amplifiers (PAs). The proposed network structure is directly compared with alternative neural network solutions, which have been successfully employed for digital predistortion (DPD). The operation of this network is validated for DPD with experimental measurements with wideband signals using the latest generation of commercially available RF hardware. The novel network structure proposed in this work is demonstrated, in practice, to have better performance for a normalized mean square error (NMSE), an adjacent channel leakage ratio (ACLR), and an error vector magnitude (EVM) compared with the most popular previously published neural network.

Index Terms—Digital predistortion (DPD), machine learning, neural networks, power amplifiers (PAs).

I. INTRODUCTION

The fields of wireless communications and AI technology are improving at breakneck speeds and with the roll out of the fifth generation (5G) of wireless systems comes the promise of lower latency, higher capacity, as well as increased data throughput speeds [1]. These high-level promises bring about a great deal of complications with the hardware used for 5G systems.

An important component in wireless communications is the power amplifier (PA). The PA, in conjunction with an antenna, allows a signal to be transmitted for long distances with high reliability. Operating the PA more efficiently can be achieved by increasing the output signal power level or employing a higher efficiency PA design; however, this can create nonlinear distortions in the signal caused by the gain compression and memory effects introduced by the PA. These distortions cause spectral regrowth, which brings about an undesired increase in the bandwidth of the signal [2].

Techniques were developed to get the best of both worlds in the compromise between the efficiency and the signal integrity. Digital predistortion (DPD) techniques based on a plethora of mathematical models have been proposed to counter nonlinear distortions. These techniques distort the original signal before it reaches the PA, such that after the amplification, it retains the linearity of the original signal. Many implementations of DPD based on polynomial structures and artificial neural networks (ANNs) have been proposed. These techniques strive to improve the linearization performance while retaining low computational complexity.

This article proposes a novel ANN predistorter based on the real-value time-delayed neural network (RVTDDNN) [3]. This novel DPD model replaces the standard activation function in the ANN with a spline-based activation function layer with trainable parameters. This technique has been shown to be more effective than standard activation functions for tasks using deep neural networks and are often called deep splines [4]. The name deep splines implies the use of a deep neural network, so from this point, it will be called the segmented spline curve (SSC) to better describe its functionality in this article. The SSC activation function is comparable in computational complexity to the rectified linear unit (ReLU). Although using an ReLU activation function with an RVTDDNN instead of the hyperbolic tangent is possible, the performance of such a solution requires a large number of coefficients to be viable [5]. In general, using a hyperbolic tangent function enables the creation of an RVTDDNN for DPD with reasonable performance and a lower number of coefficients. This poses an issue with the hardware implementation of the RVTDDNN, as the computation of the hyperbolic tangent activation function requires divisions and exponents, which require complex algorithms. Such algorithms are generally iterative by nature and, thus, can have several times higher latency than operations, such as addition or multiplication [6].

Excluding the traditional approaches, such as the RVTDDNN, there have been numerous artificial intelligence-based techniques proposed for PA behavioral modeling and DPD. Earlier techniques used for PA behavioral modeling include a prior knowledge time-delayed neural network (PKTDNN) [7] or...
global recurrent neural networks (GRNNs) [8]. While the performance of these techniques is promising, additional validation would be required to show their viability for DPD. More recently, approaches, such as the vector decomposed time-delayed neural network (VTDNN) [9] and the vector decomposed long short-term memory (VLDSTM) [10], have been proposed and shown to have good performance. These approaches, while well performing, require additional high-latency operations on top of what is required for RVTDNN. These techniques decompose the IQ vector to its magnitude and phase components before the running the data through the ANN model and recombine them after the ANN model. This requires iterative algorithms, such as coordinate rotation digital computer (CORDIC), which, similar to divisions, has higher latency compared with simple addition, subtraction, and multiplication operations [11]. In addition, to get odd-order magnitude terms requires the computation of the square root, which is also an iterative, high latency operation [12].

From this point forward, the enhanced ANN architecture will be called the SSC time-delayed neural network (SSCTDNN). The SSCTDNN makes a trade-off between the memory required and the computational complexity of the ANN. The SSCTDNN requires a substantially higher number of total coefficients than a regular RVTDNN, and this is due to the SSC requiring multiple coefficients for each neuron in the ANN. However, a distinction must be made between the total number of coefficients and active coefficients. Active coefficients are coefficients that are accessed during the inference of the SSCTDNN, where inference in this case means to run the data through the ANN model and predistorting the signal. Along with the standard weights and biases accessed in an ANN, the ones accessed at inference time in the SSCTDNN include two coefficients per neuron in layers that use the SSC activation function. The computational complexity during inference is lower compared with training, as the ANN model is trained using a batch of samples at a time, and thus, most, if not all, of the spline coefficients are used and updated at once. This means that even if a major part of the coefficients of the SSCTDNN are in the activation functions, the number of coefficients used during inference time is the same or lower compared with the RVTDNN while achieving significantly better DPD performance.

The remainder of this article is laid out as follows. In Section II, the RVTDNN and the SSCTDNN are compared, and the difference between their activation functions is explained. Next, in Section III, the proposed technique is compared against the current state-of-the-art solutions through experimental results. Finally, in Section IV, the conclusions are drawn from the results of this experimental investigation.

II. NEURAL NETWORK STRUCTURES

A. RVTDNN and ARVTDNN

One of the most common ANN structures used for PA behavioral modeling and DPD is the RVTDNN. The RVTDNN has the real-valued in-phase (I) and quadrature samples (Q), along with their delayed taps as inputs and the predistorted I and Q samples as outputs, as shown in Fig. 1 [3]. In addition, to improve modeling performance, envelope terms can be added as inputs, which can be seen as the gray inputs into the ANN model. The technique with these additional inputs is known as the augmented RVTDNN (ARVTDNN) [13]. Although not a part of the original techniques, both RVTDNN and ARVTDNN can benefit from having linear inputs directly into the output layer of the model, as it improves modeling performance [14]. These inputs are shown in gray below the hidden layer neurons in Fig. 1. Aside from the inputs and outputs of the RVTDNN and the ARVTDNN, the neurons in the hidden layer have bias terms $B$ that grant an additional layer of flexibility when the network is trained.

In general, the complexity of a neural network is judged by the number of coefficients. The number of coefficients for a single hidden layer RVTDNN can be found by the following expression:

$$N_c = ((M + 1) \times 2) \times N + N + 2 \times N + 4$$

(1)

where $M$ is the memory depth, and $N$ is the number of neurons in the hidden layer. Here, $((M + 1) \times 2) \times N$ represents the number of weights in between the input layer and the hidden layer, the standalone $N$ represents the number of bias coefficients $B$, and $2 \times N$ represents the number of weights between the hidden layer and the output layer of the RVTDNN. Finally, the additional four coefficients at the end represent the linear I and Q input coefficients. For the number of coefficients of the ARVTDNN, the number of envelope terms $E$ is inserted into the calculation, and the modified equation finally becomes

$$N_c = ((M + 1) \times (2 + E)) \times N + N + 2 \times N + 4.$$  

(2)

B. Activation Functions for RVTDNN and ARVTDNN

1) Rectified Linear Unit Activation Function: The ReLU is a piecewise linear function. This function outputs the same value as the input if it is positive and zero in other cases, as shown in Fig. 2. ReLU is the preferred activation
in a lot of cases, as it does not have vanishing gradients [15] and is simple to compute. It can be expressed as follows:

$$f(x) = \begin{cases} x, & \text{for } x \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$  \hspace{1cm} (3)$$

2) Hyperbolic Tangent Sigmoid Activation Function: The hyperbolic tangent sigmoid (Tanh) is a nonlinear activation function. It is the most common activation function used for the feedforward ANN-based PA behavioral models and predistorters due to it providing the best performance for such tasks [13]. This function is continuous, and the output tends toward negative one or positive one depending on the input. An issue with sigmoid-based functions is that they have vanishing gradients, which makes training more difficult and slower [15]. This function can be seen in Fig. 3 and can be expressed with the following formula:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{2}{1 + e^{-2x}} - 1.$$  \hspace{1cm} (4)$$

C. SSCTDNN

While the base structure of the SSCTDNN is similar to the RVTDNN and the ARVTDNN, there are two key differences. The first of these differences is the lack of bias terms in the hidden layer. This is due to the coefficients present in the SSC layer, as they can, during training, infer their own bias without needing extra coefficients. Second, the envelope terms used in the ARVT DNN model do not provide significant (if any) benefit in modeling with the SSCTDNN due to the flexibility of the SSC layer and, thus, do not need to be included. The structure of the SSCTDNN is shown in Fig. 4.

D. Spline Activation Function

The spline activation function is a segmented activation function that, after training, approximates any linear or nonlinear function using linear spline segments. This function has an array of trainable parameters $C$, which has length $L$. Before training, these coefficients can be initialized with different values to approximate different activation functions, such as the ReLU, Tanh, or any other function. The curve shown in Fig. 5 is an example of the SSC activation function that has been trained for any one neuron in the ANN. The green stars represent the trained coefficients, and the blue line is the piecewise linear interpolation. This function allows for a better approximation of the functions required for DPD, as it can converge to any linear and nonlinear functions depending
on the amount of segments in the spline. This is the main advantage over the standard activation functions, as ANNs using sigmoid-based functions require multiple neurons to approximate a similar curve. The evaluation of the SSC can be expressed as follows:

$$f(x) = \left( C[s + 1] - C[s] \right) \frac{x + \frac{1}{\delta} - s}{\delta} + C[s]$$  \hspace{1cm} (5)$$

where $x$-axis width of a spline segment is $\delta = 2/(L - 1)$, and the segment index $s$ can be found with the following equation:

$$s(x) = \left\lceil \frac{x + \frac{1}{\delta}}{\delta} \right\rceil.$$  \hspace{1cm} (6)

Although these formulas appear more complex, the computations within can be broken down to only multiplications, additions, and subtractions. This is beneficial to hardware implementations, as it removes the requirement of iterative algorithms for operations, such as division. This is advantageous, as such algorithms have higher latency [6]. It is important to notice the divisions by $\delta$, which can be instead converted into multiplications by $\delta^{-1}$, as the $\delta$ variable is calculated once upon the initialization of the SSC and can be stored for further use later. In addition, if the number of segments $L$ is chosen with care, the multiplication can be reduced to a simple bit shifting operation, further reducing the latency.

With this added layer of coefficients, the total coefficient complexity for the SSCTDNN can be calculated as follows:

$$N_c = ((M + 1) \times 2) \times N + L \times N + 2 \times N + 4.$$  \hspace{1cm} (7)

The main difference of the computational complexity of the SSCTDNN and the RVTDNN is the SSC layer coefficients $C \times N$. It is important to note that not all of these coefficients will be used during the inference of the SSCTDNN. The number of coefficients used during inference is calculated as follows:

$$N_{c} = ((M + 1) \times 2) \times N + 2 \times N + 2 \times N + 4.$$  \hspace{1cm} (8)

### E. Training the SSCTDNN

Choosing a training architecture to train SSCTDNN is very important. The main techniques used for DPD training are direct learning architecture (DLA), indirect learning architecture (ILA), and occasionally iterative learning control (ILC).

The advantage of using DLA or ILC algorithms is the resistance to noise in your training dataset [16], which is desirable, as, in noisy scenarios, there is measurable improvement over ILA. However, these improvements come with a disadvantage in terms of complexity. Additional computation is required to achieve the perfect signal before any training for ANN occurs when using ILC [17], whereas in the case of DLA, the additional complexity can go as far as requiring a separate PA model ANN. This ANN, after it is trained, is used as a source of error gradients for training the DPD model [18], which significantly increases the amount of computation required. Thus, the training for the SSCTDNN is done using ILA to reduce training complexity. The ILA training is shown in Fig. 6. This training begins with modeling the inverse PA to achieve the perfect signal before any training for ANN occurs or until further iterations provide little to no performance benefit. The training for most ANN-based techniques, including SSCTDNN, is done through backpropagation. It is important to note that backpropagation relies on the gradients produced by the functions. In the case of the SSC layer, the gradients for each of the spline functions can be reduced to the slope of the piecewise linear spline segment as follows:

$$\frac{\partial f}{\partial x} = \frac{C[s + 1] - C[s]}{\delta},$$  \hspace{1cm} (9)

### III. EXPERIMENTAL VALIDATION

#### A. Hardware and Software Setup

The hardware setup used for the experimental validation contains a Xilinx ZC706 FPGA board with an Analog Devices ADRV-DPD1/PCBZ (ADRV-DPD1) transceiver board. The ADRV-DPD1 contains the PA that is being linearized, which, in this case, is the Skyworks SKY66297-11 4-W PA. The ADRV-DPD1 board connects to a 30-dB attenuator, which is then connected to a Rohde & Schwarz FSL spectrum analyzer for observation. The hardware bench setup is shown in Fig. 7. The neural networks are implemented using Python with the PyTorch machine learning library. The data for training are preprocessed in MATLAB, and the transmission and reception of the PA signals are done with the Analog Devices AD9375 small cell evaluation software (SCES). The block diagram for both the hardware and the software is shown in Fig. 8. The signal is transmitted at a center frequency of 2.655 GHz. The signal used was a 40-MHz long-term evolution (LTE) signal with peak-cancellation crest factor reduction (PC-CFR) with a peak-to-average power ratio (PAPR) of 7.5 dB. It is important to note that the proposed solution is viable for any center frequency, as training is done at baseband frequencies. In addition, higher bandwidth signals can also be used if the hardware supports higher sampling rates for transmission and reception of the signal.

#### B. ANN Parameters

The ANN training parameters for the SSCTDNN and the RVTDNN are set as follows. The optimizer used is
the adaptive moment estimation (Adam) optimizer. For the ANN-based techniques, the learning rate is set to 0.01, a batch size of 3000 with a total of 6000 samples, the number of epochs is set to 1000, and the memory depth is set to 2. For the memory polynomial (MP), the order was set to 5 with three memory taps achieving 20 complex (40 real) coefficients. All of the techniques were trained for three ILA iterations. The values of neurons were chosen to make the tested models have a similar number of neurons. The performance of ANN predistorters is proportional to the number of coefficients [5] and, thus, should be the most fair comparison of the tested ANN techniques. It is important to note that the techniques could also be compared in terms of the number of operations; however, this would require an in-depth study of operation counts for each technique and is outside the scope of this article. Two separate test cases were carried out for the SSCTDNN; the first case had the number of neurons set to 9, and for the second case, this number was set to 19. In both of the SSCTDNN cases, the number of SSC segments was set to 9. The number of neurons for the RVTDNN is set to 17. Finally, for the ARVTDNN, the number of neurons was set to 13 with a first-order envelope term as an additional input. The two separate cases for the SSCTDNN aim to directly compare the performance with a similar total amount of coefficients as the opposing techniques, while the second case aims to compare the performance of SSCTDNN with a matching number of active coefficients. The hyperparameters of the ANN-based and the MP models were chosen, such that they achieve good performance while maintaining reasonable complexity.

C. Results

Table I shows the normalized mean square error (NMSE), adjacent channel leakage ratio (ACLR), error vector magnitude (EVM), and the number of coefficients for each technique. The two separate cases for the SSCTDNN are distinguished by the number of neurons each one has in the brackets. In addition, in the coefficients column of Table I, the total amount of coefficients is shown for all techniques, while the active coefficients of the SSCTDNN are displayed in brackets. It is apparent that out of all of the cases tested, the proposed techniques come out on top across all of the measured performance metrics. It can be seen that the nine-neuron SSCTDNN provides an increase in performance while also using less than half of the coefficients compared with both ARVTDNN and RVTDNN during inference time of the ANN. In the case of 19-neuron SSCTDNN, the performance increase is even more apparent, with over 3-dB improvement in the ACLR compared with the ARVTDNN while maintaining a slightly lower number of coefficients used during its inference time. Comparing with the results of the MP model, it can be seen that the novel model once again has greater performance in all three metrics. The clear improvement in performance compared with
the state of the art is also shown in the spectrum analyzer capture, which can be seen in Fig. 9. The frequency domain representation shown in Fig. 9 is consistent with the results acquired and shows the level of nonlinearity reduction for all of the techniques tested.

IV. CONCLUSION

Neural networks have been firmly established as a viable solution for DPD. Early neural networks are themselves considered capable of universal approximation; however, in this article, it has been shown that if the traditional structures and building blocks of these networks are reimagined, a more accurate and computationally efficient solution is possible for DPD. The proposed network structure has been bench marked with two popular and effective neural network solutions and one polynomial-based solution for DPD. These traditional structures are constrained by the rigid building blocks, and as a result, their performance suffers. The network solution proposed in this article is experimentally validated to outperform these traditional solutions. The improvement is quantified in terms of NMSE, ACPR, and EVM, in fair comparisons maintaining the same number of coefficients in each solution.

ACKNOWLEDGMENT

The authors would like to thank Bill O’Brien for his valuable input on discussions about neural networks.

REFERENCES

[1] M. Agiwal, A. Roy, and N. Saxena, “Next generation 5G wireless networks: A comprehensive survey,” IEEE Commun. Surveys Tuts., vol. 18, no. 3, pp. 1617–1655, 3rd Quart., 2016.
[2] F. M. Ghanouchi and O. Hammi, “Behavioral modeling and predistortion,” IEEE Microw. Mag., vol. 10, no. 7, pp. 52–64, Dec. 2009.
[3] S. Boumaiza and F. Mkadem, “Wideband RF power amplifier predistortion using real-valued time-delay neural networks,” in Proc. 39th Eur. Microw. Conf. (EuMC), 2009, pp. 1449–1452.
[4] P. Bohra, J. Campos, H. Gupta, S. Aziznejad, and M. Unser, “Learning activation functions in deep (spline) neural networks,” IEEE Open J. Signal Process., vol. 1, pp. 295–309, 2020.
[5] R. Hongyo, Y. Egashira, T. M. Hone, and K. Yamaguchi, “Deep neural network-based digital predistorter for Doherty power amplifiers,” IEEE Microw. Wireless Compon. Lett., vol. 29, no. 2, pp. 146–148, Feb. 2019.
[6] S. F. Oberman and M. J. Flynn, “An analysis of division algorithms and implementations,” IEEE Trans. Comput., vol. 46, pp. 833–854, 1995, [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.30.1308
[7] J. Dooley, B. O’Brien, and T. J. Brazil, “Prior knowledge input neural network for microwave power amplifier modeling,” in Proc. Int. Workshop Integr. Nonlinear Microw. Millim-Wave Circuits, 2006, pp. 156–159.
[8] B. O’Brien, J. Dooley, and T. J. Brazil, “RF power amplifier behavioral modeling using a globally recurrent neural network,” in IEEE MTT-S Int. Microw. Symp. Dig., Jun. 2006, pp. 1089–1092.
[9] Y. Zhang, Y. Li, F. Liu, and A. Zhu, “Vector decomposition based time-delay neural network behavioral model for digital predistortion of RF power amplifiers,” IEEE Access, vol. 7, pp. 91559–91568, 2019.
[10] H. Li, Y. Zhang, G. Li, and F. Liu, “Vector decomposed long short-term memory model for behavioral modeling and digital predistortion for wideband RF power amplifiers,” IEEE Access, vol. 8, pp. 63780–63789, 2020.

Andrius Vaicaitis

John Dooley (Member, IEEE) received the B.E. degree (Hons.) in electronic engineering with communications from Maynooth University, Maynooth, Ireland, in 2019. He is currently pursuing the Ph.D. degree in telecommunications hardware and network optimization at Maynooth University, Maynooth, Ireland, under Science Foundation Ireland ADVANCE CRT funding. Mr. Vaicaitis was awarded for the highest grades in his graduating class.

Andrius Vaicaitis (Student Member, IEEE) received the B.E. degree (Hons.) in electronic engineering with communications from Maynooth University, Maynooth, Ireland, in 2019. He is currently pursuing the Ph.D. degree in telecommunications hardware and network optimization at Maynooth University, Maynooth, Ireland, under Science Foundation Ireland ADVANCE CRT funding. Mr. Vaicaitis was awarded for the highest grades in his graduating class.