On the Exploration of Promising Analog IC Designs via Artificial Neural Networks

Nuno Lourenço, João Rosa, Ricardo Martins, Helena Aidos, António Canelas, Ricardo Póvoa, Nuno Horta
Instituto de Telecomunicações
Instituto Superior Técnico, Universidade de Lisboa
Lisboa, Portugal
{nlourenco, ricmartins, haidos, acanelas, rpovoa, nuno.horta}@lx.it.pt

***Abstract*—In this paper, deep learning and artificial neural networks (ANNs) are used to size analog integrated circuits. In classical optimization-based sizing strategies the computational intelligent techniques are used to iterate over the map from devices sizes to circuits' performances, provided by design equations or circuit simulations, whereas here, it is performed an exploratory work on how ANNs can be capable of solving analog integrated circuit sizing as a direct map from specifications to the sizing. The proposed methodology was implemented and tested on a real circuit topology, with promising results. Moreover, trained ANNs were able to extend the circuit performance boundaries outside the train/validation set, showing that more than a mapping for the training data, the model is capable of learning reusable design patterns and provide promising designs.**

Keywords—Analog IC Sizing, Data Mining, Artificial Neural Networks, Deep Learning

I. Introduction

While integrated circuits (IC) are mostly implemented using digital circuitry, analog and radio-frequency (RF) circuits are still necessary and irreplaceable in the implementation of most interfaces and transceivers. However, unlike the digital design where an automated flow is established for most design stages, the absence of effective and established computer-aided-design (CAD) tools for electronic design automation (EDA) of analog and RF IC blocks poses the largest contribution to their bulky development cycles, leading to long, iterative and error-prone designer intervention over their entire design flow [1].

The prevalent method to analyse and evaluate analog ICs is focused on the mapping from the devices' sizes to the circuit's performance figures. This mapping can be done using approximate equations, which usually support manual design approaches, or, using the circuit simulator, that is used to verify and fine tune the manual design. For automatic sizing, simulation-based optimization is the most prevalent method in both industrial [2][3] and academic [4][5] environments.

Hence, instead of going from the target specification to the corresponding device sizes, the designer or the EDA tool are actually going from tentative devices' sizes to the corresponding circuit performance countless times, i.e., trying combinations of design variables to find a sizing such that the circuit meets specifications, as shown in Fig.1 (a), even if similar design were already done.

In this work we address this important issue, by performing an exploratory research on how artificial neural networks (ANNs) and deep learning [6] can solve the circuit sizing problems directly, as shown in Fig. 1 (b), given the appropriate training set.

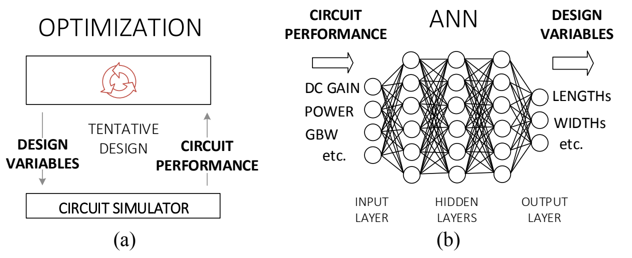

Fig. 1. Illustration of how different methods address the analog IC sizing problem, and, find the dimensions of the devices so that the circuit meet the specifications. (a) Optimization-based sizing: Inverse approach; (b) ANNs: Direct approach.

ANNs were already used in some EDA methodologies for analog IC sizing [7][8], but to replace/complement the circuit simulator in the optimization-based sizing approach of Fig. 1 (a). While the evaluation speed is greatly increased by avoiding time-consuming circuit simulations, the accuracy lost is only recovered by returning to the circuit simulator in later stages of the optimization. Moreover, training these models is done over the entire design space, which spends valuable resources modelling and evaluation large regions of unusable design combinations. In [9] this aspect is somewhat addressed, but still, the ANNs were trained to replace the simulator, instead of being trained to size the circuit for a given specification.

In this work, we explore how an ANN that is trained using circuit sizing solutions from previous optimizations can learn the design patterns of the circuit, and, as shown in Section IV, can even generate sizing for efficient circuits on specifications outside those in the training dataset.

The remainder of this paper is organized as follows. In Section II, analog IC sizing is formulated as a machine learning problem and the dataset is defined. In Section III, the proposed multi-objective sizing optimization, and also, ANN models, including data pre-processing, normalization, model and hyperparameter tuning, training, and, finally, a method to sample analog IC sizing from the trained ANNs, are described. In Section IV, the proposed method is used in the sizing of an analog amplifier, and, finally, Section V draws the conclusions of this paper.

This work is funded by FCT/MEC through national funds and when applicable co-funded by FEDER – PT2020 partnership agreement under the project UID/EEA/50008/2013, and by the Fundação para a Ciência e Tecnologia (Grants: SFRH/BPD/104648/2014, SFRH/BPD/103127/2014, SFRH/BPD/120009/2016, SFRH/BD/133662/2017, SFRH/BD/103337/2014).

978-1-5386-5153-7/18/$31.00 ©2018 IEEE

II. Problem and Dataset Definition

Let $V_{<i>} \in \mathbb{R}^N$ be the vector of design variables that define the sizing of the circuit (e.g. widths and lengths of the CMOS devices), where the index *i* inside the chevron identifies solution point *i* in the dataset, and; $S_{<i>} \in \mathbb{R}^D$ be the vector of the corresponding circuit performance figures (e.g. Power consumption, DC Gain). The ANNs presented in this work were trained to predict the most likely sizing given the target specifications, as shown in (1).

$$V = \operatorname{argmax}(P(V|S)) \tag{1}$$

Hence to train the ANN, the training data is comprised of a set *T*, of *M* data pairs $\{V, S\}_{<i>}$. Since we want the model to learn how to design circuits properly, these pairs must correspond to useful designs, e.g., optimal or quasi-optimal design solutions for any given sizing problems.

A. Inequality Specifications and Data Augmentation

While the previous definition allows training a model suitable for analog IC sizing, circuits' target specifications are, more often than not, defined as inequalities instead of equalities. Therefore, an ANN trained to map $S \rightarrow V$ may have difficulties extrapolating to map specification outside the train set distributions, even if the requested specification values are easily meet by designs in the training set.

In fact, if the sizing $V_{<i>}$ corresponds to a circuit whose performance is $S_{<i>}$, then it is also a valid design for any specifications whose performance targets, $S'_{<i>}$, are worse than $S_{<i>}$. Keeping this in mind, an augmented dataset, *T'* can be obtained from *T* as the union of *T* and *K* copies of *T*, as indicated in (2), where the for each copied sample *i*, the circuit true performance $S_{<i>}$ is replaced by $S'_{<i>}$ according to (3).

$$T' = \{T \cup T^{C1} \cup T^{C2} \cup \ldots \cup T^{CK}\} \tag{2}$$

$$S'_{<i>} = S_{<i>} + \Delta\Gamma\left(\frac{\gamma}{M}\sum_{j=1}^{M} S_{<j>}\right) \tag{3}$$

Where, $\gamma \in]0, 1[$ is a factor that is used to scale the average performances, Δ is a D×D diagonal matrix of random numbers between [0, 1], and, $\Gamma \in \{-1, 1\}^D$ is the target diagonal matrix. This matrix defines the scope of usefulness of the circuit. Its diagonal components are -1 for performance figures in which a smaller target for the specification is also fulfilled by the true performance of the design, e.g. a DC Gain. Oppositely, the value 1 is for the diagonal components for the performance figures that meet specifications that are larger than the true performance of the circuit, e.g. power consumption.

III. ANN Models for Analog IC Sizing

The ANNs models considered in this work consider fully connected layers without weight sharing. Given the number of features that are used in the model and the size of the datasets that will be considered in this application, the models considered only a few (3 to 5) hidden layers. The best structure depends of the dataset, but a systematic method is proposed later in this Section to specify such models. To train and evaluate the model, the datasets are split in training (80%-90%) and validation sets (20%-10%). An additional small test set, whose specifications are drawn independently, are also used to verify the real world application of the model.

A. Polynomial Features and Normalization

To increase training efficiency, the ANN is trained with an input *X* that is the feature mapping Φ of *S* normalized. Each input data sample $X_{<i>}$ is given by:

$$X_{<i>} = \frac{\Phi(S_{<i>}) - \mu_\Phi}{\sigma_\Phi} \tag{4}$$

where $\Phi(S_{<i>})$ is a second order polynomial mapping of the original specifications, e.g., for $S_{<i>}$ = [a, b, c], $\Phi(S_{<i>})$ = [a, b, c, a^2, a*b, a*c, b^2, b*c, c^2]; μ_Φ is the mean of the components of Φ, and, σ_Φ is the standard deviation of the components of Φ. The output of the network, *Y*, is defined from *V* by (5):

$$Y_{<i>} = \frac{V_{<i>} - \min(V)}{\max(V) - \min(V)} \tag{5}$$

B. Model Structure and Hyper parameter tuning

In terms of the ANN structure, 3 to 5 hidden layers, with 30 to 240 nodes each, showed to be effective. In terms of non-linearity both the Sigmoid and ReLU [10] showed good results, with a small advantage to the later. The guidelines that were used to select the hyperparameters for the ANNs shown in Section IV were as follow:

- The number of nodes in each layer increases in the first layers to create a rich encoding, and then, decreases toward the output layer to decode the predicted circuit sizing;
- The models were first designed to have high performance (low error) in the training data, even if overfitting was observed. This method allowed to determine the minimum complexity that can model the training data, overfitting was then addressed using L2 weight regularization, as shown in Fig. 2;
- Finally, grid search is done over the hyperparameters (number of layer, number of nodes per layer, non-linearity and regularization factor) to fine tune the model.

C. Training

The loss function, *L,* of the model that is optimized during training is the mean squared error (MSE) of the predicted outputs *Y'* with respect to the true *Y* plus the *L2* norm of the model's weights, *W*, times the regularization factor λ, according to (6).

$$L = \frac{1}{M}\sum_{j=1}^{M}\left(\left(Y'_{<j>} - Y_{<j>}\right)^T\left(Y'_{<j>} - Y_{<j>}\right)\right) + \lambda\|W\|^2 \tag{6}$$

The training of the models is done using Adam [11], a variant of stochastic steepest descent with both adaptive learning rate and momentum that provides good performance, moreover, it is quite robust with respect to its hyper parameters.

Other error metrics such as mean absolute error (MAE) are also considered when validating the results.

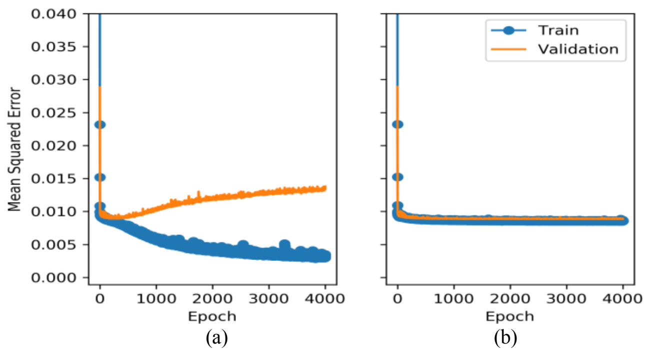

Fig. 2. Evolution of prediction error on train and validation sets during training: (a) ANN that overffits the training data, showing high error on the validation set. (b) Same ANN trained with *L2* norm weigh regularization, showing better performance on the validation set.

D. *Sampling from the ANN*

Sampling from the ANN is done using (3) *P* times, with *Γ* replaced by *–Γ*, i.e., we ask the model to predict a set of *P* sizing solutions given circuit performances that are better than the desired specifications. If some performance figures, from those used when training the model, are not specified, the corresponding component in the diagonal random matrix Δ from (3) should be a random value in the range of [-1, 1].

For example, if the target specifications are gain bandwidth product (GBW) over 30 MHz and current consumption (I_{DD}) under 300 µA, and, the model was trained with DC Gain, GBW and I_{DD}. A set of circuit performances given to the ANN could be, e.g., {(50dB, 35MHz, 290µA), (75dB, 30MHz, 285µA), (60dB, 37MHz, 250µA), …, (90dB, 39MHz, 210 µA)}.

The reasoning behind this sampling is: even if the ANN has properly learned the designs patterns present in the performances of the sizing solutions in the training data, when the performance trade-off implied by the target specifications being requested are not from the same distribution than the training data, the prediction of the ANN can be strongly badly biased. While using the augmented dataset described in Section II.B alleviates this bias, it is still better to sample the ANN this way.

The selection of solutions from the *P* predictions of the ANN is done by simulating the predicted circuit sizing, and, either using a single value metric, such a Figure-of-Merit (FoM) for the circuit, select the most suitable solution, or, using some sort of Pareto dominance to present a set of solutions exploring the trade-off between specifications.

IV. Case Study

For proof of concept, the amplifier using voltage combiners for gain enhancement (VCOTA) from [12] was considered. To illustrate the problem difficulty, 1 million random designs were generated and simulated, and, not even one design yield a set of modest target performances (all devices with saturation and overdrive voltages larger than 50 mV, FoM > 900 MHz·pF/mA, GBW > 30 MHz, DC Gain > 40 dB). The circuit's schematic is shown in Fig. 3, showing the devices with annotated design variables.

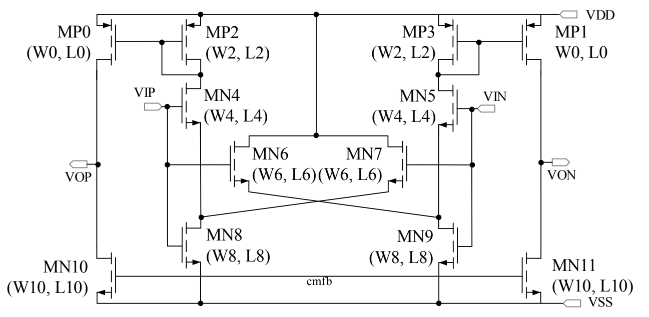

Fig. 3. Circuit schematic showing the devices and corresponding design variables (channel width: W's, and, channel length: L's).

A. *Dataset*

For this example, the dataset before any augmentation has 16,600 different design points. This dataset was obtained from a series of previously done studies on this circuit for the UMC 130 nm technology design process, and, contains only optimized circuit sizing solutions. The circuit performances that were considered to train the ANN were DC Gain, I_{DD}, GBW and Phase margin (PM), and, the ranges of values found in the dataset are shown in Table I.

TABLE I. Performance Ranges in the DataSet

	DC Gain	**GBW**	**I_{DD}**	**PM**
Max	56.8 dB	78 MHz	395 uA	80º
Min	44.7 dB	34 MHz	221 uA	60º

B. *ANNs Structure and Training*

Three ANNs were trained for this circuit, i.e., ANN-1, ANN-2, and ANN-3. The structure considered has 14 input variables (obtained from the second order polynomial feature extension of the 4 performance figures from Table I), 3 hidden layers with 120, 240, 60 nodes each, and, the output layer has 12 nodes, the width and lengths of the 6 matched transistor pairs. The non-linearity used in all nodes was ReLU. All ANNs were implemented in Python with TensorFlow [13-17], training was done on four Intel I7 cores@2.6GHz.

ANN-1, was trained on the original dataset, for 5000 epochs (passes through the entire dataset) with batches of 512 samples. Its training took less than 15 minutes. ANN-2, was trained on the dataset augmented 40 times (almost 700K samples) for the same 5000 epochs. Its training took approximately 8 hours. ANN-3 was also trained on the same augmented dataset, but only for 500 epoch, but was initialized with weights from ANN-1. Its training took less than an hour, when adding the 15 mins from ANN-1, the total training time was around one hour. Their performance after training on the training and validation sets is summarized in Table II.

TABLE II. Performance of Trained ANNs

	MSE Train	**MSE Val.**	**MAE Train**	**MAE Val.**
ANN-1	0.0159	0.0157	0.0775	0.0776
ANN-2	0.0124	0.0123	0.0755	0.0750
ANN-3	0.0124	0.0124	0.0754	0.0753

C. Sampling the ANNs for new designs

Sampling the ANNs was done as described in Section III-D, with P = 100 and $\gamma = 0.15$, i.e., 100 random samples with a deviation of up to 15% from the specifications. Selection of the best solution was done by FoM for target 1, GBW for target 2, and I_{DD} and FoM for target 3. It is easily seen, by the performance of the obtained circuits, that the ANNs learned the design patterns and can even extrapolate for specifications outside those of the training data. Moreover, circuits with FoMs larger than 1000 were obtained in all samplings of the ANNs. For each target the ANNs were sampled only once (P times), for target 3 multiple points (rows) are listed to show how the ANNs tried to meet the impossible specifications in different ways. ANN-1 was able to find a solution with a very low value for I_{DD}, however the corresponding value for GBW is an order of magnitude smaller than the target. To show that ANN-1 could also find more meaningful solutions around the target GBW, an additional solution with GBW = 30 MHz is also presented in Table III. ANN-1 can explore easily new specifications, but generates greater variability and worse designs when sampled. ANN-2 and ANN-3 are more stable generating always good designs when sampled inside the training data. ANN-2 shows more limitations when trying to explore new specifications, at this time we can only speculate that the data augmentation procedure alone may camouflage the true design patterns. ANN-3, because it used transfer learning from ANN-1, is more flexible to new specifications, but still lags when compared to ANN-1.

TABLE III. PERFORMANCE OF SAMPLED DESIGNS

	#HF[a]	DC Gain	GBW	I_{DD}	PM	FoM[b]
Target 1		50 dB	60 MHz	300 uA	65°	
ANN-1	0.33	50 dB	63 MHz	318 uA	64°	1180
ANN-2	1	51 dB	61 MHz	320 uA	65°	1153
ANN-3	1	51 dB	63 MHz	325 uA	65°	1165
Target 2		40 dB	150 MHz	700 uA	55°	
ANN-1	0.24	44 dB	148 MHz	822 uA	54°	1082
ANN-2	0.21	49 dB	60 MHz	325 uA	73°	1106
ANN-3	1	43 dB	100 MHz	509 uA	61°	1182
Target 3		50 dB	30 MHz	150 uA	65°	
ANN-1 [c]		50 dB	3 MHz	141 uA	74°	116
ANN-1	0.73	50 dB	30 MHz	205 uA	74°	889
ANN-1 [d]		49 dB	67 MHz	329 uA	60°	1215
ANN-2 [c]	1	54 dB	38 MHz	240 uA	71°	950
ANN-2 [d]		54 dB	46 MHz	268 uA	64°	1033
ANN-3 [c]	0.97	55 dB	30 MHz	217 uA	69°	842
ANN-3 [d]		54 dB	54 MHz	309 uA	56°	1050

[a] Ratio of the number of solutions with FoM higher than 850 (the minimum value in the training data was 900) to the total number of samples; [b] MHz·pF/mA; [c] Best I_{DD}; [d] Best FoM. The presented solutions all meet the constraints on overdrives and saturation with margins larger than 45 mV.

V. CONCLUSIONS

In this work, deep learning methodologies were used to develop ANNs that successfully predicted analog IC sizing for an amplifier, given their intended target performances. This is a disruptive work, as no such approach has been taken in the field of analog and RF IC sizing, showing that a properly trained ANN can learn design patterns and generate circuit sizing that are correct for specification trade-offs, including those not provided in the training data.

Clarifying, the purpose of this paper was not to propose a complete automation solution for the analog IC sizing, as this works only scratches the surface of the impact the ANNs and deep learning may have in analog CAD and EDA. There are still several opportunities where deep learning and ANN might improve analog EDA. A great possibility, and, at the same time one of the most challenging issues, is how to collect enough data to train more complex models. Given the importance of data, both in terms of quantity and quality. Data collection and aggregation is a great opportunity to the EDA community to define intra and inter-organizational protocols and formats, facilitating the creation of rich and meaningful datasets that can potentially enable automatic analog design to reuse design patterns instead of the specific design solutions.

REFERENCES

[1] N. Lourenço, R. Martins, and N. Horta, "Automatic Analog IC Sizing and Optimization Constrained with PVT Corners and Layout Effects," Springer, 2017. Hardcover ISBN 978-3-319-42036-3

[2] Cadence, "Virtuoso Analog Design Environment GXL". Retrieved from http://www.cadence.com, March, 2018.

[3] Mentor, a Siemens Business, "Eldo Platform". Retrieved from https://www.mentor.com/products/ic_nanometer_design/analog-mixed-signal-verification/eldo-platform, March 2018.

[4] N. Lourenço, et al., "AIDA:Layout-aware analog circuit level sizing with in-loop layout generation", *Integration, the VLSI Journal.* 55(9):316-329, 2016.

[5] R. González-Echevarría, et al., "An Automated Design Methodology of RF Circuits by Using Pareto-Optimal Fronts of EM-Simulated Inductors," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 36, no. 1, pp. 15-26, Jan. 2017.

[6] Goodfellow, I., Bengio, Y., and, Courville, A. Deep Learning. MIT Press. 2016.

[7] G. Alpaydin, S. Balkir and G. Dundar, "An evolutionary approach to automatic synthesis of high-performance analog integrated circuits," in IEEE Transactions on Evolutionary Computation, vol. 7, no. 3, pp. 240-252, June 2003. doi: 10.1109/TEVC.2003.808914J.

[8] G. Wolfe and R. Vemuri, "Extraction and use of neural network models in automated synthesis of operational amplifiers," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 22, no. 2, pp. 198-212, Feb. 2003.doi: 10.1109/TCAD.2002.806600M.

[9] Hongzhou Liu, A. Singhee, R. A. Rutenbar and L. R. Carley, "Remembrance of circuits past: macromodeling by data mining in large analog design spaces," Proceedings 2002 Design Automation Conference (IEEE Cat. No.02CH37324), 2002, pp. 437-442. doi: 10.1109/DAC.2002.1012665

[10] V. Nair and G. E. Hinton. "Rectified linear units improve restricted boltzmann machines". In Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML'10), 2010.

[11] D. P. Kingma, J. Ba, "Adam: A Method for Stochastic Optimization". In: CoRR, abs/1412.6980, 2014.

[12] R. Povoa; N. Lourenco; R. Martins; A. Canelas; N. Horta; J. Goes, "Single-Stage Amplifier biased by Voltage-Combiners with Gain and Energy-Efficiency Enhancement," in IEEE Transactions on Circuits and Systems II: Express Briefs , doi: 10.1109/TCSII.2017.2686586

[13] M. Abadi, et al. "TensorFlow: Large-scale machine learning on heterogeneous systems", 2015. Software available from tensorflow.org.

[14] Chollet, F., et al., "Keras". GitHub, 2015.

[15] Fabian Pedregosa, et al. "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research, 12, 2825-2830 (2011)

[16] John D. Hunter. Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90-95 (2007), DOI:10.1109/MCSE.2007.55

[17] Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37