




Article

Differentiable Constraints' Encoding for Gradient-Based Analog Integrated Circuit Placement Optimization

António Gusmão ^{1,2}, Pedro Alves ^{1,2}, Nuno Horta ^{1,2}, Nuno Lourenço ^{1,3} and Ricardo Martins ^{1,2,*}¹ Instituto de Telecomunicações, Universidade de Lisboa, 1049-001 Lisboa, Portugal² Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisboa, Portugal³ Departamento de Informática, Universidade de Évora, 7005-869 Évora, Portugal

* Correspondence: ricmartins@lx.it.pt

Abstract: Analog IC design is characterized by non-systematic re-design iterations, often requiring partial or complete layout re-design. The layout task usually starts with device placement, where the several performance figures and constraints to be met escalate its complexity immensely, and, due to the inherent tradeoffs, an “optimal” floorplan solution does not usually exist. Deep learning models are now establishing for the automation of the placement task of analog integrated circuit layout design, promising to bypass the limitations of existing approaches based on: time-consuming optimization processes with several constraints; or placement retargeting from legacy designs/templates, which rely heavily on legacy layout data. However, as the complexity of analog design cases tackled by these methodologies increases, a broader set of topological constraints must be supported to cover the different layout styles and circuit classes. Here, model-independent differentiable encodings for regularity, boundary, proximity, and symmetry island constraints are formulated for the first time in the literature, and an unsupervised loss function is used for the artificial neural network model to learn how to generate placements that follow them. The use of a deep learning model makes push-button speed placement generation possible, additionally, as only sizing data are required for its training, it discards the need to acquire legacy layouts containing insights into this vast set of, often neglected, constraints. The model is ultimately used to produce floorplans from scratch at push-button speed for real state-of-the-art analog structures, including technology nodes not used for training. A case-study comparison with a floorplan design made by a human-expert presents improvements in the fulfillment of every constraint, reaching an overall improvement of around 70%, demonstrating the approach's value in placement design.

Keywords: automatic placement; deep learning; electronic design automation; physical synthesis; topological constraints



Citation: Gusmão, A.; Alves, P.; Horta, N.; Lourenço, N.; Martins, R. Differentiable Constraints' Encoding for Gradient-Based Analog Integrated Circuit Placement Optimization.

Electronics **2023**, *12*, 110. <https://doi.org/10.3390/electronics12010110>

Academic Editors:

Costas Psychalinos and
Esteban Tlelo-Cuautle

Received: 14 November 2022

Revised: 9 December 2022

Accepted: 23 December 2022

Published: 27 December 2022



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A vast number of techniques for automatic analog integrated circuit (IC) layout synthesis have been proposed in the past decades, which are mostly based on: (1) *optimization processes with several topological constraints* [1,2] or (2) *placement retargeting from legacy designs* [3,4] or *templates* [5,6]. In the first, every solution is generated from scratch in a, generally, time-consuming optimization process where several metrics (e.g., area, wiring topology estimates) are minimized while fulfilling the constraints. Still, even though this process moves toward a global optimum solution, it does not necessarily mean that a meaningful layout solution has been found for the circuit designer. In the second, the solution being generated matches its structure with those found on a library (or more frequently, a single instance) of legacy layouts/templates. While the solution is quickly produced from the guidelines of validated data, these methodologies struggle to generalize beyond the legacy data available, which is often scarce and may not necessarily correspond to the exact topology being generated. Nonetheless, the limitations of these approaches

have been identified for long, which have hampered their widespread propagation across established commercial computer-aided design tools for analog IC design.

However, tremendous advances in machine learning (ML), and more specifically, deep learning (DL), in the past decade, are offering new alternatives to the electronic design automation (EDA) of electronic circuits [7,8]. Similarly, for analog IC placement, DL-based approaches were recently proposed [9–12], attempting to match the fast generation abilities of placement retargeting with the flexibility of optimization. Nonetheless, as the complexity of real-world analog design cases tackled by these methodologies increases, a wider set of topological constraints must be supported to cover multiple layout styles and circuit classes, e.g., device symmetry or SIs, proximity, preplaced, regularity, boundary, separation, current/signal-flows, among others [13]. Therefore, the research described in this manuscript follows the most recent trials on analog placement automation by using a non-linear unsupervised artificial neural network (ANN) model, trained with thousands of sizing solutions which can be generated via established automatic methods such as [14,15], where innovative differentiable encodings for regularity, boundary, proximity, and symmetry island (SI) are introduced for the first time in the literature. Ultimately, the model supports different circuit topologies on its input layer and outputs at push-button speed the placement coordinates of each device for any given sizing and constraints, as shown in Figure 1.

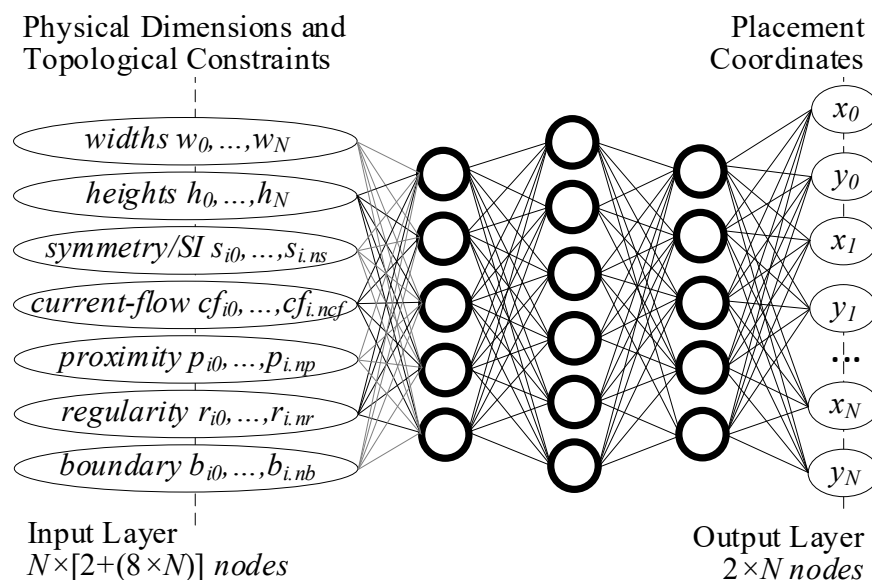


Figure 1. ANN architecture used to solve the map from the physical dimensions and topological constraints to the placement coordinates. N represents the maximum number of circuit devices supported by the model.

This document is organized as follows. Section 2 provides the related work and contributions. Section 3 details the input vector features, constraints’ encodings, and loss function. Finally, Sections 4 and 5 address the results and conclusions.

2. Related Work and Contributions

Some of the recent ML applications for analog IC layout include building block recognition from circuit netlists in order to build the circuit’s hierarchy automatically [16], generation of wells during layout design that imitate human-designer choices [17], or even assistance/guidance for traditional path-finding routing algorithms [18]. In the particular case of analog IC placement, a straightforward fully supervised ANN that reproduces the layout patterns of a dataset of legacy placement solutions was proposed [10]. In [11], it was enhanced, providing a multi-circuit model that distinguishes between topologies by describing the topological constraints at the ANN’s input layer. Still, both models

are highly dependent on the description of the circuits' traits on the model's input layer, preventing them from scaling for circuits with a larger number of devices than those found in the dataset. Therefore, an attention-based graph-to-sequence model was recently proposed, where the encoder-decoder architecture is scalable and invariant to the order in which the devices are provided at the input layer. These unsupervised placement techniques have proved to compete or outperform those generated by state-of-the-art optimization-based placement approaches in [12]. Note that the DL models produced these results in milliseconds, while the optimization techniques took from several seconds up to several minutes, thus, establishing the DL approach as a competitive alternative to the more traditional methods. Nonetheless, due to the intricate formulations required for efficient training of a DL model, only fundamental topological constraints were effectively supported, i.e., symmetry, proximity, and current-flows.

Supporting a broader set of topological constraints is mandatory to tackle a more comprehensive set of industry-standard analog IC design cases and produce aesthetic-wise solutions for the circuit designer. However, their implementation in DL-based tools is not trivial. Therefore, the major contributions of this work are:

- In this paper, differentiable implementations of boundary, regularity, and SI constraints are formulated for the first time in the literature. These are inherently model-independent, i.e., they can be used to train simple models, such as the multilayer perceptron [11], or even complex encoder-decoder architectures [12];
- Existing automatic approaches for analog IC placement that focus on *retargeting from legacy designs/templates* [3,4], or even ML-based approaches [10], use previously designed placement solutions (layouts) that comprise expert design knowledge. These examples are used as a surrogate for the explicit definition of the constraints to be met. However, legacy layouts containing robust implementations of the abovementioned constraints are scarce and expensive to obtain, and thus, here, the model's training requires only sizing data, which is way easier to produce, and the model is focused on complying with the explicitly and efficiently described topological constraints. When compared with *optimization processes with several topological constraints* [1,2], which also explore the possibility of explicitly defining the constraints to be met, the inherent time-consuming optimization cycles are bypassed by the use of deep models. This is, once the model is fully trained, it can produce several valid solutions for a singular problem at push-button speed through its generative characteristics;
- The novel formulations for the boundary, regularity, proximity, and SI constraints are used to train a model that produces placement solutions from scratch at push-button speed for several state-of-the-art block-level analog IC structures, including circuit topologies and technology nodes not used for training, ultimately proving its generalization capabilities.

3. Unsupervised ANN Model with a Broader Topological Constraints Coverage

The general architecture of the proposed ANN model was previously illustrated in Figure 1.

3.1. Notation

Table 1 introduces the notation used through this document.

Table 1. Notation.

Notation	Definition
N	Number of devices in the model
$\mathbf{A}_{r \times c}$	An $r \times c$ matrix
$\mathbf{A}_{r \times c \times l}$	An $r \times c \times l$ matrix
\mathbf{v}_r	An $r \times 1$ column vector
$A_{i,j}$	Entry i, j of matrix \mathbf{A}

Table 1. Cont.

Notation	Definition
v_i	i^{th} entry of vector \mathbf{v}
\mathbf{I}_n	$n \times n$ identity matrix
$\mathbf{1}_{n \times m}$	$n \times m$ matrix filled with ones
$diag(\mathbf{v}_r)$	Diagonal matrix $\mathbf{D}_{r \times r}$ where each entry in the diagonal is $D_{i,i} = v_i \forall_{i \in \{1, \dots, r\}}$ and zero otherwise
$diag(\mathbf{A}_{r \times r})$	Column vector \mathbf{d}_r obtained from the entries on the main diagonal of the square matrix $\mathbf{A}_{r \times r}$, $d_i = A_{i,i} \forall_{i \in \{1, \dots, r\}}$
\mathbf{A}^T	Transpose of a two-dimensional matrix $\mathbf{A}_{r \times c}$. $\mathbf{A}^T = \begin{bmatrix} A_{1,1} & \cdots & A_{r,1} \\ \vdots & \ddots & \vdots \\ A_{1,c} & \cdots & A_{r,c} \end{bmatrix}_{c \times r}$
$\Delta^u(\mathbf{A})$	Upper triangular matrix $\mathbf{T}_{r \times r}$ obtained from the entries on and above the main diagonal of the square matrix $\mathbf{A}_{r \times r}$, $T_{i,j} = A_{i,j} \forall_{i \leq j; i, j \in \{1, \dots, r\}}$ and zero otherwise
$\Sigma^n(\mathbf{A})$	Matrix obtained from summing the values of the multi-dimensional matrix \mathbf{A} (dimensionality greater or equal to 2) along dimension n .
$\Sigma^{n,m}(\mathbf{A})$	Matrix obtained from summing the values of the multi-dimensional matrix \mathbf{A} (dimensionality greater or equal to 3) first along dimension n then m .
$\Sigma(\mathbf{A})$	Sum of all elements of \mathbf{A}
$\mathbf{M}^n(\mathbf{A})$	Compute the maximum of the multi-dimensional matrix \mathbf{A} over dimension n
$\mathbf{m}^n(\mathbf{A})$	Compute the minimum of the multi-dimensional matrix \mathbf{A} over dimension n
$\mathbf{m}(k, \mathbf{A}), \mathbf{M}(k, \mathbf{A})$	Compute a multi-dimensional matrix with the dimension of \mathbf{A} whose entries are respectively, the minimum or maximum of each value of \mathbf{A} and k .
$\mathbf{m}(\mathbf{A}, \mathbf{B}), \mathbf{M}(\mathbf{A}, \mathbf{B})$	A multi-dimensional matrix with the dimension of \mathbf{A} and \mathbf{B} whose entries are the respectively, the minimum or maximum of each value of \mathbf{A} and \mathbf{B} .
$\mathbf{A} \circ \mathbf{B}$	Hadamard (or Schur) product of two multi-dimensional matrices with the same dimension. Example for two dimensional matrices: $\mathbf{A}_{r \times c} \circ \mathbf{B}_{r \times c} = \begin{bmatrix} A_{1,1}B_{1,1} & \cdots & A_{1,c}B_{1,c} \\ \vdots & \ddots & \vdots \\ A_{r,1}B_{r,1} & \cdots & A_{r,c}B_{r,c} \end{bmatrix}_{r \times c}$
\mathbf{a}^T, \mathbf{b}	Inner product of two vectors. $\langle (\mathbf{a}_{r \times 1})^T, \mathbf{b}_{r \times 1} \rangle = \Sigma^1(\mathbf{a} \circ \mathbf{b})$
$\mathbf{A} \otimes \mathbf{B}$	Kronecker product of two multi-dimensional matrices with the same number of dimensions. Example for two-dimensional matrices: $\mathbf{A}_{r \times c} \otimes \mathbf{B}_{s \times d} = \begin{bmatrix} A_{1,1}\mathbf{B} & \cdots & A_{1,c}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{r,1}\mathbf{B} & \cdots & A_{r,c}\mathbf{B} \end{bmatrix}_{rs \times cd}$

3.2. Input Vector Features

As in its baseline implementation, the model inputs the scaled width/height of the physical implementation (bounding box dimensions) of each device composing the circuit, i.e., a vector of size $2N$, where N is the number of devices within that circuit topology, resulting from flattening the device size dimension matrix $\mathbf{D}_{N \times 2}$. Additionally, as thoroughly described in [11], symmetry (that restricts two devices to a mirrored placement along a symmetry axis (SA)), proximity (that describes groups of devices that should be placed closely), and current-flow (that force a monotonic path between the different devices composing the ‘current-flow’) requirements are passed to the ANN by three $N \times N$ unweighted adjacency matrices: \mathbf{S}^T , \mathbf{P} , and \mathbf{C} respectively. This work proposes the use of five more matrices to encode the boundary ($\mathbf{B}_{N \times N}^x, \mathbf{B}_{N \times N}^y$), regularity ($\mathbf{R}_{N \times N}^x, \mathbf{R}_{N \times N}^y$), and SI constraints ($\mathbf{S}_{N \times N}^g$). While the sets of topological constraints used for the circuit topologies under study in this work were specified by experienced analog

designers, these can be obtained directly by using works that automate their extraction process. These works can be either based on deterministic approaches [19,20], or, as developed recently, by using graph-based ML techniques [16,21,22]. Given the circuit's sizing and the different constraint graphs, the model is trained to output a placement that fulfills the specified constraints. The placement is output as a $2N$ long vector, which can be interpreted as an $N \times 2$ matrix \mathbf{L} , whose columns correspond to each devices' coordinates in the x and y axis respectively.

3.3. Boundary

This constraint leads to easier routing access to the interfaces and fewer parasitic interactions, as shown in Figure 2b. A device with a boundary constraint must be placed between the remaining devices of the circuit and the "virtual" enclosing shape that contains all the devices, and, since this verification must be made in one of the four directions, the boundary error is divided into two axes, x and y . The matrices $\mathbf{B}^x_{N \times N}$ and $\mathbf{B}^y_{N \times N}$ encode the boundary constraint graphs for the x and y axis respectively. Figure 3a,b show the \mathbf{B}^x and \mathbf{B}^y matrix respectively for the circuit shown in Figure 2.

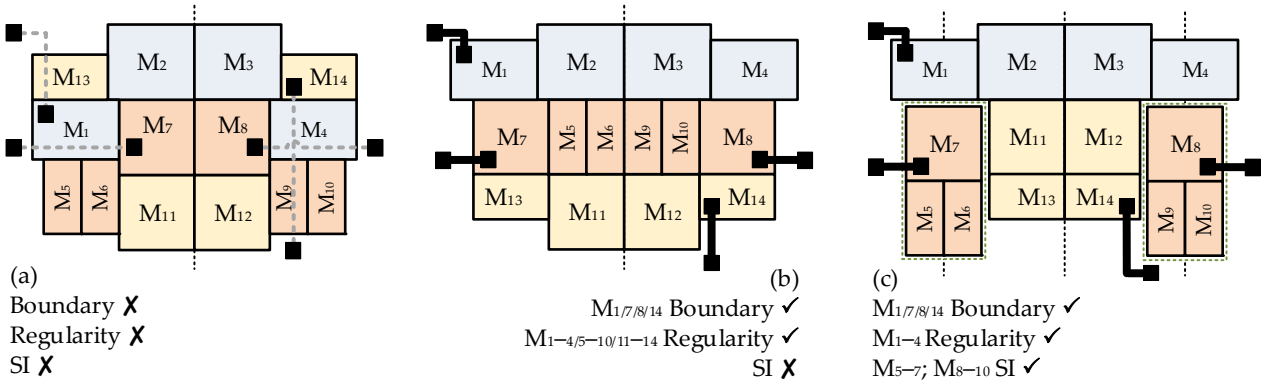


Figure 2. Three symmetric placements of the generic circuit: (a) Devices connected to the input/output ports are placed inside the circuit layout; (b) devices are placed within the circuit boundary and three horizontal regularity constraints are implemented; (c) two symmetry islands are formed.

In practice, the \mathbf{B}^x and \mathbf{B}^y matrices encode an ordering along each axis, where $B^x_{i,j} = 1$ encodes that device i 's maximum x coordinate should be greater than j 's maximum x coordinate, and an analogous meaning is given to $B^y_{i,j}$. On the other hand, $B^x_{i,j} = -1$ encodes that device i 's minimum x coordinate should be lesser than j 's minimum x coordinate, and an analogous meaning is given to $B^y_{i,j}$. Thus, if device i should be on the top boundary, then $B^y_{i,k} = 1, \forall k \in [1, N]$. Or if device i should be on the left boundary, then $B^x_{i,k} = -1, \forall k \in [1, N]$.

Thus, to estimate the boundary error relative to the y axis, the *TopTop* and *BottomBottom* errors between each device pair must be considered, each respectively represented by $\mathbf{B}^{tt}_{N \times N}$ and $\mathbf{B}^{bb}_{N \times N}$, calculated through:

$$\mathbf{B}^{tt/bb} = M\left(0, M(\mathbf{B}^y, 0) \circ \left[\mathbf{1}_{N \times 1} \otimes (\mathbf{y}^{+/-})^T - \mathbf{1}_{1 \times N} \otimes \mathbf{y}^{+/-} \right] \right) \quad (1)$$

where \mathbf{y}^-_N corresponds to the vector containing the minimum y coordinate of each device in the evaluated placement, defined as $y_i = L_{i,2}, \forall i \in [1, N]$, and \mathbf{y}^+_N corresponds to the maximum y coordinate vector defined as $y_i^+ = L_{i,2} + D_{i,2}, \forall i \in [1, N]$.

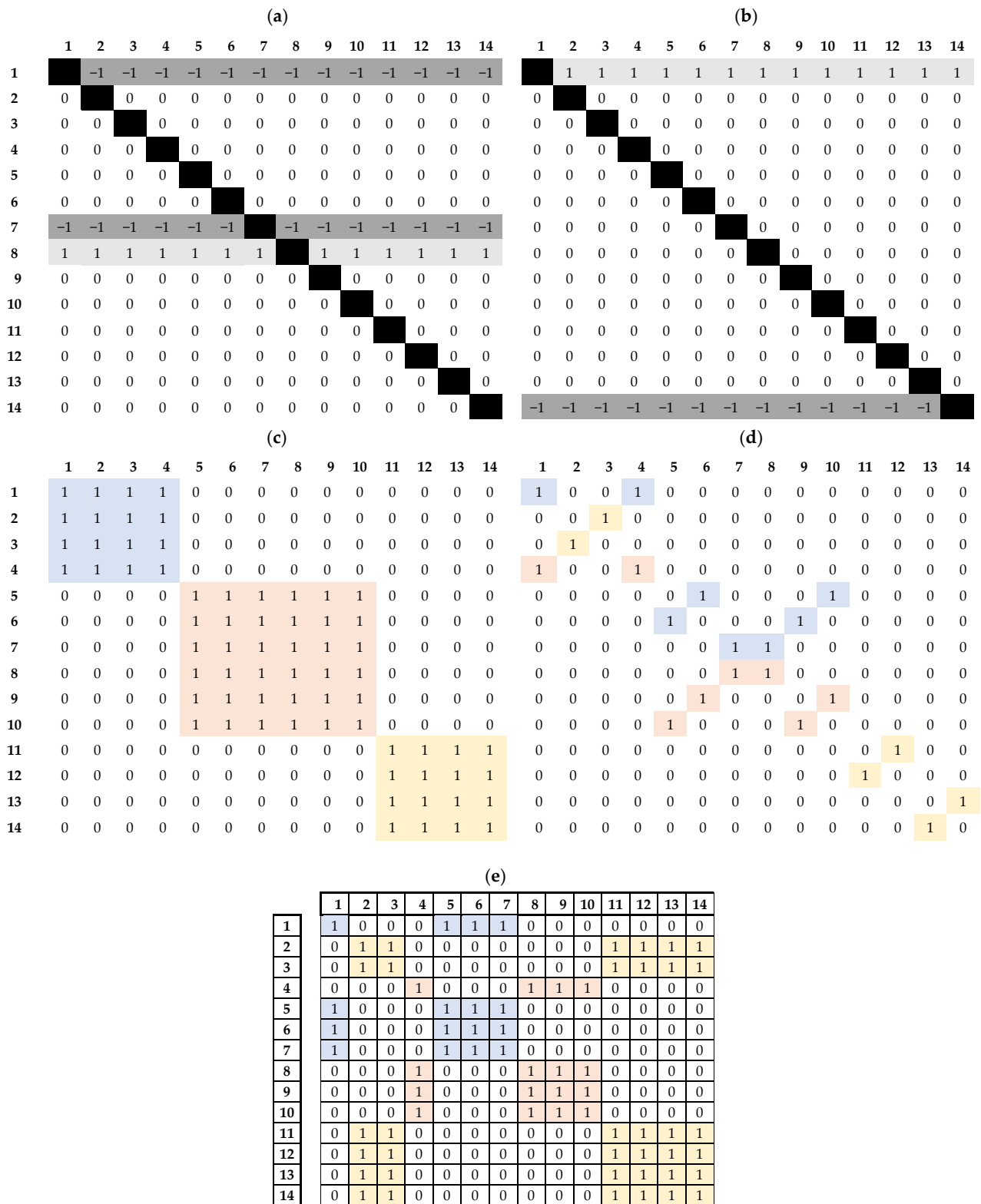


Figure 3. One-hot encoded matrices used to provide: (a) boundary constraints in the x axis (Figure 2b); (b) boundary constraints in the y axis (Figure 2b); (c) row regularity constraints (Figure 2b); (d) symmetry relations (Figure 2c); and (e) symmetry island grouping (Figure 2c) to the deep model.

Similarly, for the x axis, the *RightRight* and *LeftLeft* errors between each device pair, represented by $\mathbf{B}_{N \times N}^{rr}$ and $\mathbf{B}_{N \times N}^{ll}$ respectively, are given by:

$$\mathbf{B}^{rr/ll} = M\left(0, M(\mathbf{B}^x, 0) \circ \left[\mathbf{1}_{N \times 1} \otimes (\mathbf{x}^{+/-})^T - \mathbf{1}_{N \times N} \otimes \mathbf{x}^{+/-} \right] \right) \quad (2)$$

where \mathbf{x}_N^- corresponds to the vector containing the minimum x coordinate of each device in the evaluated placement, defined as $x_i = L_{i,1}, \forall i \in [1, N]$, and \mathbf{x}_N^+ corresponds to the maximum x coordinate vector defined as $x_i^+ = L_{i,1} + D_{i,1}, \forall i \in [1, N]$.

Finally, another factor must be considered to maximize the boundary constraint's effectiveness: whether a device is located between a boundary-constrained device and its objective (the placement's boundary). This can be estimated by measuring the overlap between the devices' projections in the axis perpendicular to the corresponding boundary axis for every device pair with an ordering error greater than zero (i.e., $B_{i,j}^{tt} > 0$ or $B_{i,j}^{bb} > 0$ or $B_{i,j}^{rr} > 0$ or $B_{i,j}^{ll} > 0$, since this means that device j is closer to the objective boundary than i , and it must be checked whether it is blocking device i 's path).

To measure the overlap between each device pair's projections, consider the distance that separates the opposing faces of the two rectangles. For example, matrix $\mathbf{dX}_{N \times N}^{+-}$ measures, for every pair of devices i and j , the distance between device i 's maximum x coordinate face and device j 's minimum x coordinate face. Resulting in four $N \times N$ matrices, two for each axis: \mathbf{dX}^{+-} , \mathbf{dX}^{-+} , \mathbf{dY}^{+-} and \mathbf{dY}^{-+} :

$$\mathbf{dX}^{+-/-+} = \left| \mathbf{1}_{1 \times N} \otimes \mathbf{x}^{+/-} - \mathbf{1}_{N \times 1} \otimes (\mathbf{x}^{-/+})^T \right| \quad (3)$$

$$\mathbf{dY}^{+-/-+} = \left| \mathbf{1}_{1 \times N} \otimes \mathbf{y}^{+/-} - \mathbf{1}_{N \times 1} \otimes (\mathbf{y}^{-/+})^T \right| \quad (4)$$

The matrices $\mathbf{dX}_{N \times N}^o$ and $\mathbf{dY}_{N \times N}^o$ that define the measured distance to solve the overlap in each axis are given through:

$$\mathbf{dX}^o = \frac{1}{\mathbf{1}_{N \times 1} \otimes \mathbf{w}^T} (\mathbf{Sx}^{+-} \mathbf{dX}^{+-} + \mathbf{Sx}^{-+} \mathbf{dX}^{-+}) \quad (5)$$

$$\mathbf{dY}^o = \frac{1}{\mathbf{1}_{N \times 1} \otimes \mathbf{h}^T} (\mathbf{Sy}^{+-} \mathbf{dY}^{+-} + \mathbf{Sy}^{-+} \mathbf{dY}^{-+}) \quad (6)$$

where \mathbf{h}_N is the column vector containing the devices' heights given by $h_i = D_{i,2}, i \in [1, N]$, \mathbf{w}_N is the column vector containing the devices' widths given by $w_i = D_{i,1}, i \in [1, N]$ and the four $N \times N$ matrices \mathbf{Sx}^{+-} , \mathbf{Sx}^{-+} , \mathbf{Sy}^{+-} , and \mathbf{Sy}^{-+} are scaling weights that function as a soft minimum for the distance between the devices' opposing faces in each direction, and are defined by:

$$\mathbf{Sx}^{+-} = \frac{\mathbf{dX}^{-+}}{\mathbf{dX}^{-+} + \mathbf{dX}^{+-}} \quad (7)$$

$$\mathbf{Sx}^{-+} = \mathbf{1}_{N \times N} - \mathbf{Sx}^{+-} \quad (8)$$

$$\mathbf{Sy}^{+-} = \frac{\mathbf{dY}^{-+}}{\mathbf{dY}^{-+} + \mathbf{dY}^{+-}} \quad (9)$$

$$\mathbf{Sy}^{-+} = \mathbf{1}_{N \times N} - \mathbf{Sy}^{+-} \quad (10)$$

The use of these scaling weights was introduced in [11], where it is explained in detail, and results in an output with a value between the minimum term and the maximum term, but, proportionally closer to the minimum term, such that when the minimum term approaches 0, its weight approaches 1.

Thus, given \mathbf{dX}^0 and \mathbf{dY}^0 , the two matrices, $\mathbf{B}_{N \times N}^h$ and $\mathbf{B}_{N \times N}^v$ that evaluate how much devices stand in the way of the boundary constrained devices are given by:

$$\mathbf{B}^h = \mathbf{M} \left(0, \frac{\mathbf{B}^{tt} + \mathbf{B}^{bb}}{|\mathbf{B}^{tt} + \mathbf{B}^{bb}|} \right) \circ \mathbf{dX}^0 \tag{11}$$

$$\mathbf{B}^v = \mathbf{M} \left(0, \frac{\mathbf{B}^{rr} + \mathbf{B}^{ll}}{|\mathbf{B}^{rr} + \mathbf{B}^{ll}|} \right) \circ \mathbf{dY}^0 \tag{12}$$

The inclusion of these two errors results from a gradient-aware approach since the objective of the boundary error is not only to identify situations where the constraint is fulfilled but also to, given two distinct placements, rank them faithfully to a designer’s preferences. Locally, this results in defining the function’s gradients and information regarding how to update a placement to correct its errors intelligently. In particular, this added overlap error results in a gradient that pushes devices away from the path of the boundary-constrained devices, resulting in cooperative behavior. Figure 4 shows an example placement where the device D_1 should be placed in the top boundary, resulting in three distinct errors $B_{1,2}^{tt}$, $B_{1,3}^{tt}$, and $B_{1,2}^h$. Note that no error exists between D_1 and D_4 despite the existing horizontal overlap between them since there is no $B_{1,4}^{tt}$ error, thus the max function from Equation (11) returns 0 and $B_{1,4}^h = 0$. Intuitively, D_4 does not stand between D_1 and its objective (the top boundary), thus, it should not be pushed away.

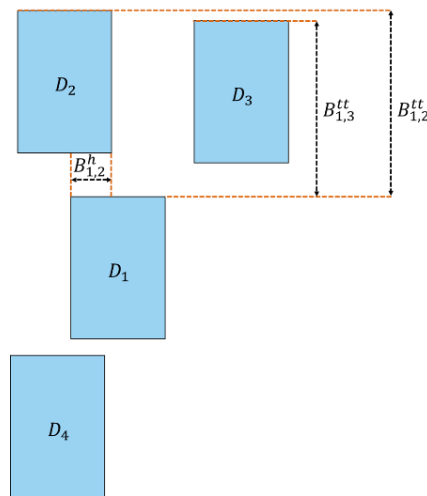


Figure 4. Example placement where device D_1 should be placed in the top boundary, resulting in three distinct errors $B_{1,2}^{tt}$, $B_{1,3}^{tt}$, and $B_{1,2}^h$. Note that no error exists between D_1 and D_4 despite the existing horizontal overlap between them since D_4 is not closer to the top boundary than D_1 .

The pairwise boundary error $\ddot{\mathbf{B}}_{N \times N}$ is calculated via:

$$\ddot{\mathbf{B}} = \mathbf{B}^{tt} + \mathbf{B}^{bb} + \mathbf{B}^{rr} + \mathbf{B}^{ll} + \mathbf{B}^h + \mathbf{B}^v \tag{13}$$

Finally, this pairwise error could then be properly normalized and pooled through the methods discussed in [12], resulting in a final error B .

3.4. Regularity

This constraint arranges the devices into regular layout structures such as rows and columns, often resulting in a compact and meaningful layout, as shown in Figure 2b. The regularity constraint is split into two constraints, one for each axis, x and y . The regularity constraint applied to the x axis constrains device pairs to be vertically aligned, i.e., their

centroids should have the same x coordinate. Whereas the y regularity constrains the pair’s centroids to have the same y coordinate. For a given axis, the regularity axis between every pair of devices can be determined by calculating the pair’s centroid’s x and y position through $\hat{\mathbf{X}}_{N \times N}$ and $\hat{\mathbf{Y}}_{N \times N}$:

$$\hat{\mathbf{X}} = \frac{1}{2} \left(\left(\mathbf{x}^- + \frac{\mathbf{w}}{2} \right)^T \otimes \mathbf{1}_{N \times 1} + \left(\mathbf{x}^- + \frac{\mathbf{w}}{2} \right) \otimes \mathbf{1}_{1 \times N} \right) \quad (14)$$

$$\hat{\mathbf{Y}} = \frac{1}{2} \left(\left(\mathbf{y}^- + \frac{\mathbf{h}}{2} \right)^T \otimes \mathbf{1}_{N \times 1} + \left(\mathbf{y}^- + \frac{\mathbf{h}}{2} \right) \otimes \mathbf{1}_{1 \times N} \right) \quad (15)$$

where \mathbf{w}_N and \mathbf{h}_N each respectively contain each device’s width and height and are defined as $w_i = D_{i,1}$, $h_i = D_{i,2}$, $\forall i \in [1, N]$.

After calculating each regularity *column* and *row* axis between each pair of devices, given by $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$ respectively, the errors $\mathbf{dR}^x_{N \times N}$ and $\mathbf{dR}^y_{N \times N}$ that measure the deviation between each device to its correspondent *column* and *row* axis per pair is calculated, using:

$$\mathbf{dR}^x = \mathbf{R}^x \circ \left(\left(\mathbf{x}^- + \frac{\mathbf{w}}{2} \right)^T \otimes \mathbf{1}_{N \times 1} - \hat{\mathbf{X}} \right)^2 \quad (16)$$

$$\mathbf{dR}^y = \mathbf{R}^y \circ \left(\left(\mathbf{y}^- + \frac{\mathbf{h}}{2} \right)^T \otimes \mathbf{1}_{N \times 1} - \hat{\mathbf{Y}} \right)^2 \quad (17)$$

where $\mathbf{R}^x_{N \times N}$ and $\mathbf{R}^y_{N \times N}$ are the adjacency matrices of the graphs that encode the *column* and *row* regularity constraints respectively. These graphs encode, for a pair of devices i and j , whether they should be aligned over a vertical or horizontal axis, respectively. Figure 3c shows the row regularity constraint graph’s adjacency matrix for the placement in Figure 2b.

Note how the regularity errors calculated in (16) and (17) measure, for each device pair i and j , their distance to the pair’s centroid given by $\hat{X}_{i,j}$ or $\hat{Y}_{i,j}$. The alternative would be to directly calculate the distance between each device’s centroid through $\left(x_i^- + \frac{w_i}{2} - x_j^- - \frac{w_j}{2} \right)^2$. However, experimentally, the used errors proved to be more stable. For the same placement, the alternative errors will measure a higher value than the centroid error proposed. Thus, considering the use of momentum-based optimization algorithms (e.g., Adam), the devices will build more momentum and are more likely to overshoot their target and oscillate around the minimum. Whereas the centroid approach presents a common goal for both devices, a “compromise”. That way, the devices will not build as much momentum and are more likely to stabilize around the minimum.

Given the constraint’s pairwise x and y errors, the final pairwise regularity error $\ddot{\mathbf{R}}_{N \times N}$ is given by:

$$\ddot{\mathbf{R}} = \mathbf{dR}^x + \mathbf{dR}^y \quad (18)$$

Finally, this pairwise error can then be normalized appropriately and pooled through the methods proposed in [12], resulting in the final constraint error R .

3.5. Relative Proximity

In the boundary error, the overlap error components from Equations (5) and (6) were proposed to make unconstrained devices aware of the constrained devices’ objectives, inducing a collaborative behavior. Similarly, in order to encourage the model to group devices in the same symmetry group in an island (i.e., there is a contiguous shape that can contain all devices in the same symmetry group while not containing any device outside of the group), the implementation of the proximity constraint proposed in [11,12] was updated to behave in the same cooperative manner and used in the SI error to promote island formation. Additionally, note how in past works [12], all of the symmetry, current-

flow, and overlap constraints can be satisfied, i.e., their error can be 0. The same, however, is unlikely to happen to the proximity error since the pairwise proximity error is only zero if the two devices are touching each other. This is an unnecessarily strict constraint that actively promotes overlap. A possible approach to turn this constraint satisfiable is to define a maximum threshold distance, under which the proximity error is zero, however, the size of this threshold would be, in practice, a hyperparameter to be optimized, and consequently, subject to overfitting.

Instead, a different approach was taken: the definition of proximity group was changed to a relative sense instead of an absolute one. In this implementation, the devices that form a proximity group (symmetry groups share proximity constraints as well, meaning that all symmetry groups are a proximity group) should be closer to each other than they are to all other devices. Thus, the measured error estimates how close, on average, device i is to all devices that do not belong to its proximity group compared to devices that do belong to the same symmetry group. To estimate the error associated to this constraint, first it is necessary to estimate the effective minimum distance (EMD) between every pair of devices, as defined in [12]. The distance measured by this metric corresponds to the minimum distance between any two points belonging to each rectangle (device). The pairwise EMD is represented by the matrix $E_{N \times N}$. Given this, the multi-dimensional matrix $E^r_{N \times N \times N}$ measures, for a triplet of devices i, j , and k , how closer is device i to j than it is to k , through:

$$E^r = (P \circ E) \otimes \mathbf{1}_{1 \times 1 \times N} - ((\mathbf{1}_{N \times N} - P) \circ E) \otimes \mathbf{1}_{1 \times N \times 1} \tag{19}$$

Note how the compared distances are those between devices in the same proximity group (represented by the term $(P \circ E) \otimes \mathbf{1}_{1 \times 1 \times N}$) with those between devices not in the same proximity group (represented by the term $((\mathbf{1}_{N \times N} - P) \circ E) \otimes \mathbf{1}_{1 \times N \times 1}$). To measure the pairwise island formation error, the relative distance matrix E^r must be first filtered for only occurrences where $E_{i,j} > E_{i,k}$, i.e., $E'_{i,j,k} > 0$, and then averaged over the number of devices with which device i does not have a proximity constraint:

$$\ddot{P} = \Sigma^3(M(E^r, 0)) \circ \frac{1}{(\Sigma^2(\mathbf{1}_{N \times N} - P))^T \otimes \mathbf{1}_{N \times 1}} \tag{20}$$

By considering a relative definition of proximity, not only does the constraint become satisfiable, but also the devices start behaving cooperatively. Instead of only pushing devices in the same proximity group close, the proximity error now pushes the devices not proximity-bounded away, opening space for the proximity-bounded devices to get close to one another. This is especially important because of the overlap error that often stops other constraints from fulfilling themselves, resulting in a local minimum around which the system oscillates. It should also be highlighted that this change comes at a computational cost, increasing the constraint's complexity from $O(N^2)$ to $O(N^3)$. Finally, and as before, this pairwise error can then be adequately normalized and pooled through the methods proposed in [12], resulting in the final constraint error P .

3.6. Symmetry Island

The SI groups symmetric devices are close to each other, minimizing their mismatch. Therefore, multiple SA are considered, as well as hierarchical symmetry, as shown in Figure 2c. The SI error is separated into two different calculations, the error related to the intra-group interactions and the error related to the axes that separate two different symmetry groups. Thus, this approach is limited to two hierarchical levels of symmetry, but unlimited symmetry axes are supported.

To identify the SI constraints, two different matrices are used: one, the symmetry relations matrix $S^r_{N \times N}$, to identify pairs of devices that are symmetric to one another in relation to any of the SA in the circuit; and a second, the symmetry group matrix $S^g_{N \times N}$, that signals whether two devices share the same SA set, where the SA set of a device is the set of SAs in relation to which the device is symmetric to another. For example, in Figure 2c,

three SAs are marked in a dashed line, these can be denominated, from the left to the right: S_1 , S_2 , and S_3 . For device M_1 , its SA distribution is the set $\{S_1, S_2\}$ since it is symmetric to itself in relation to S_1 and symmetric to M_4 in relation to S_2 . By defining the SA sets of each device, it is possible to conclude that devices M_1, M_5, M_6, M_7 all share the same SA set and thus, belong to the same symmetry group. Figure 3d,e show the two matrices \mathbf{S}^r and \mathbf{S}^g for the circuit shown in Figure 2.

Since this implementation is limited to two hierarchical levels of symmetry, a device can at most have two symmetry constraints, i.e., $M(\Sigma^2(\mathbf{S}^r)) \leq 2$. This makes it possible to separate the constraints into two levels, intra-group constraints and inter-group constraints. Intra-group constraints are identified for bounding two devices of the same symmetry group, whereas inter-group constraints bound devices in different symmetry groups.

3.6.1. Intra-Group Axis

For each device, the x coordinate of its intra-group symmetry relation is defined in the vector $\mathbf{s}_N^{\text{intra},x}$, given by:

$$\mathbf{s}^{\text{intra},x} = \frac{(\mathbf{S}^r \circ \mathbf{S}^g) + \text{diag}(\Sigma^1(\mathbf{S}^r \circ \mathbf{S}^g))}{2}, \mathbf{x}^- + 0.5\mathbf{w} \tag{21}$$

where $(\mathbf{S}^r \circ \mathbf{S}^g)$ identifies the intra-group relations. By adding the term $\text{diag}(\Sigma^1(\mathbf{S}^r \circ \mathbf{S}^g))$, a 1 is added to the main diagonal if the corresponding device has an intra-group symmetry constraint. Then the dot product adds the x coordinates of the devices' centroids, which when halved gives the pair's centroid.

It is possible to build a three-dimensional matrix $\mathbf{S}_{N \times N \times N}^{\text{intra},r}$ that for a device i , the matrix $\mathbf{S}_i^{\text{intra},r} = \mathbf{s}_i^g \otimes \mathbf{1}_{1 \times N} \circ \mathbf{S}^r$. Thus, $\mathbf{S}_{N \times N \times N}^{\text{intra},r}$ can be seen as a stack of N $\mathbf{S}_{N \times N}^r$ matrices where each layer i is filtered such that only the relations relative to the group to which device i belongs to are considered (i.e., only consider edges to and from a device in the group). It is important to define this matrix so that the number of symmetry constraints associated to each SA can be accurately calculated. One naïve approach to defining the number of constraints associated with each SA is through the \mathbf{S}^g matrix, where for each device, the number of elements in its group would be counted as the number of constraints. For example, for device M_1 , its group is defined by the set $\{M_1, M_5, M_6, M_7\}$ with cardinality 4, however, since devices M_5 and M_6 are symmetric to one another, then only 3 symmetry constraints are associated to this intra-group axis. This matrix is estimated via:

$$\mathbf{S}^{\text{intra},r} = (\mathbf{S}^r \otimes \mathbf{1}_{N \times 1 \times 1}) \circ (\mathbf{S}^g \otimes \mathbf{1}_{1 \times N \times 1}) \circ (\mathbf{S}^g \otimes \mathbf{1}_{1 \times 1 \times N}) \tag{22}$$

Now, a matrix $\mathbf{S}_{N \times N \times N}^{\text{intra},x}$ whose values are 0 if $S_{i,j,k}^{\text{intra},r} = 0$ and otherwise, its values are the x coordinates of the symmetric pair's centroid. The values of $\mathbf{s}^{\text{intra},x}$ and $\mathbf{S}^{\text{intra},r}$ are used to compute the matrix $\mathbf{S}_{N \times N \times N}^{\text{intra},x}$ through:

$$\mathbf{S}^{\text{intra},x} = \mathbf{S}^{\text{intra},r} \circ (\mathbf{s}^{\text{intra},x} \otimes \mathbf{1}_{N \times 1 \times N}) \tag{23}$$

The reference symmetry axis for each device, represented by the vector $\tilde{\mathbf{s}}_N^{\text{intra}}$, is computed through:

$$\tilde{\mathbf{s}}^{\text{intra}} = \Sigma^{3,2} \left(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{intra},x} \right) \circ \frac{1}{\Sigma^{3,2} \left(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{intra},r} \right)} \tag{24}$$

where the term $\Sigma^{3,2} \left(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{intra},x} \right)$ adds, for each layer i in $\mathbf{S}_{N \times N \times N}^{\text{intra},x}$, all values in the upper triangular region of the matrix, which corresponds to the different x coordinates of the centroids of all devices that belong to the same symmetry group as

device i . Then, the term $\Sigma^{3,2}(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{intra},r})$ computes for each device i , how many symmetry constraints are being considered to properly average out the group's centroid's x coordinate.

The pairwise error associated with the deviation of a pair's symmetry axis with its reference, represented by $\ddot{\mathbf{S}}_{N \times N}^{\text{intra},x}$, can be thus estimated via:

$$\ddot{\mathbf{S}}^{\text{intra},x} = \mathbf{1}_{1 \times N} \otimes \left(\mathbf{s}^{\text{intra},x} - \tilde{\mathbf{s}}^{\text{intra}} \right)^2 \circ \mathbf{S}^r \circ \mathbf{S}^g \tag{25}$$

The pairwise error associated to the vertical misalignment between an intra-group symmetric pair $\ddot{\mathbf{S}}_{N \times N}^{\text{intra},y}$ is given by:

$$\ddot{\mathbf{S}}^{\text{intra},y} = \left(\mathbf{y}^- \otimes \mathbf{1}_{1 \times N} \right) - \left(\left(\mathbf{y}^- \right)^T \otimes \mathbf{1}_{N \times 1} \right) \circ \mathbf{S}^r \circ \mathbf{S}^g \tag{26}$$

3.6.2. Inter-Group Axis

Similarly, the inter-group pairwise error associated to the deviation of the symmetry axis from the mean is calculated by first estimating, for each device, the x coordinate of its symmetry axis with a device from another group $\mathbf{s}_N^{\text{inter},x}$, then estimating, for each device, the symmetry constraints between devices in its group with devices from another group $\mathbf{S}_{N \times N \times N'}^{\text{inter},r}$ then calculating $\mathbf{S}_{N \times N \times N}^{\text{inter},x}$ and $\tilde{\mathbf{s}}_N^{\text{inter}}$ the same way they calculated for the intra-group constraints, and finally, calculating the error matrix $\ddot{\mathbf{S}}_{N \times N}^{\text{inter},x}$:

$$\mathbf{s}^{\text{inter},x} = \frac{[\mathbf{S}^r \circ (1 - \mathbf{S}^g)] + \text{diag}\left(\Sigma^1[\mathbf{S}^r \circ (1 - \mathbf{S}^g)]\right)}{2}, \mathbf{x}^- + 0.5\mathbf{w} \tag{27}$$

$$\mathbf{S}^{\text{inter},r} = (\mathbf{S}^r \otimes \mathbf{1}_{N \times 1 \times 1}) \circ ((1 - \mathbf{S}^g) \otimes \mathbf{1}_{1 \times N \times 1}) \circ (\mathbf{S}^g \otimes \mathbf{1}_{1 \times 1 \times N}) + (\mathbf{S}^g \otimes \mathbf{1}_{1 \times N \times 1}) \circ ((1 - \mathbf{S}^g) \otimes \mathbf{1}_{1 \times 1 \times N}) \tag{28}$$

$$\mathbf{S}^{\text{inter},x} = \mathbf{S}^{\text{inter},r} \circ \left(\mathbf{s}^{\text{inter},x} \otimes \mathbf{1}_{N \times 1 \times N} \right) \tag{29}$$

$$\tilde{\mathbf{s}}^{\text{inter}} = \Sigma^{3,2} \left(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{inter},x} \right) \circ \frac{1}{\Sigma^{3,2} \left(\Delta^u(\mathbf{1}_{N \times N}) \otimes \mathbf{1}_{N \times 1 \times 1} \circ \mathbf{S}^{\text{inter},r} \right)} \tag{30}$$

$$\ddot{\mathbf{S}}^{\text{inter},x} = (\mathbf{S}^r \circ (1 - \mathbf{S}^g)) \circ \left(\left(\mathbf{s}^{\text{inter},x} - \tilde{\mathbf{s}}^{\text{inter}} \right)^2 \otimes \mathbf{1}_{N \times 1} \right) \tag{31}$$

where in Equation (28), the term $((1 - \mathbf{S}^g) \otimes \mathbf{1}_{1 \times N \times 1}) \circ (\mathbf{S}^g \otimes \mathbf{1}_{1 \times 1 \times N})$ considers relations to outer-group devices from group devices, and the term $(\mathbf{S}^g \otimes \mathbf{1}_{1 \times N \times 1}) \circ ((1 - \mathbf{S}^g) \otimes \mathbf{1}_{1 \times 1 \times N})$ considers the opposite.

The pairwise error associated to the vertical misalignment between an inter-group symmetric pair $\ddot{\mathbf{S}}_{N \times N}^{\text{inter},y}$ is given by:

$$\ddot{\mathbf{S}}^{\text{inter},y} = \left(\mathbf{y}^- \otimes \mathbf{1}_{1 \times N} \right) - \left(\left(\mathbf{y}^- \right)^T \otimes \mathbf{1}_{N \times 1} \right) \circ \mathbf{S}^r \circ (1 - \mathbf{S}^g) \tag{32}$$

The pairwise error associated to the symmetry island constraint $\ddot{\mathbf{S}}_{N \times N}$ is given by:

$$\ddot{\mathbf{S}} = \ddot{\mathbf{S}}^{\text{intra},x} + \ddot{\mathbf{S}}^{\text{intra},y} + \ddot{\mathbf{S}}^{\text{inter},x} + \ddot{\mathbf{S}}^{\text{inter},y} \tag{33}$$

Finally, this pairwise error can then be normalized and pooled through the methods proposed in [12], resulting in the final constraint error S_I . It should be noted that devices in the same symmetry group form a clique in the proximity constraint graph, thus, these groups will tend to form islands.

3.7. Loss Function

The deep model is ultimately trained to minimize a loss function whose inputs are the placement of the devices (i.e., the model's output), the circuits' constraint graphs, and respective sizing (i.e., the model's inputs). This loss, shown in Equation (34), accounts for the satisfiability topological constraints considered, i.e., proximity (P), symmetry/SI (S_I), current-flows (C_F), boundary (B), and regularity (R). Still, two other metrics are also utilized to regulate the quality of produced layouts, i.e., overlaps and wasted area (W_A and O , respectively). The total loss is then given by the weighted sum of each of these constraint losses, where the $w_{a,pg,si,cf,o,bou,reg}$ variables in Equation (34) represent said weights. Note that each of these constraint-specific losses is calculated in function of the circuit's sizing (matrix $\mathbf{D}_{N \times 2}$), its positional constraints (specified by the graphs $\mathbf{P}_{N \times N}$, $\mathbf{S}_{N \times N}^r$, $\mathbf{S}_{N \times N}^g$, $\mathbf{C}_{N \times N}$, $\mathbf{B}_{N \times N}^x$, $\mathbf{B}_{N \times N}^y$, $\mathbf{R}_{N \times N}^x$, $\mathbf{R}_{N \times N}^y$) and the produced placement (matrix $\mathbf{L}_{N \times 2}$). Moreover, by using this unsupervised formulation, the model can be easily trained with unlabeled data, reinforcing its generalized application.

$$loss = w_a W_A + w_{pg} P + w_{si} S_I + w_{cf} C_F + w_o O + w_{bou} B + w_{reg} R \quad (34)$$

4. Experimental Results

The methodology was coded in Python using PyTorch ML library, and demonstrated with a dataset containing sizing solutions (split into training, validation, and test) for different state-of-the-art amplifiers on two foundries/nodes: the voltage-combiners (VC)-biased amplifier [23] (VCB, umc130), the folded VC-biased amplifier [24] (FVC, umc130, and tsmc65), and the VC-biased single-stage operational transconductance amplifier [25] (VCOTA, umc130), as shown in Table 2. Note that FVC in tsmc65 node was kept for validation and test only. The largest circuit is the VCOTA with 22 devices, whose schematic is illustrated in Figure 5. For it, 11 symmetry pairs were set by an experienced IC designer, 12 current-flows, 6 boundary constraints, 5 regularity constraints (row only), and 15 proximity constraints, as detailed in Table 3. This procedure was performed for the remaining amplifiers, with VCB being the only case study with SIs. Additionally, to balance the dataset, the sizing solutions of FVC were augmented by shuffling the order of the devices. To place larger analog circuit/system topologies, this work can be directly applied to each of the sub-blocks of the system, which, when placed, would be assembled at the top level given a set of topological constraints. As an alternative, the methodology could be used to place the entire complete mixed-signal or system in one single step, the major drawback being the deep model's training which would escalate proportionally.

The model's input vector has size 3916, and the weights of Equation (34) were empirically chosen and set to: $w_a = 0.1$, $w_s = 95$, $w_{cf} = 80$, $w_{bou} = 20$, $w_{reg} = 2$, $w_o = 90$, and $w_{pg} = 25$. The ANN model comprises 5 hidden layers (with 5000/2500/1000/250/100 neurons each) and 1 output layer (44 nodes). The model was trained for 1000 epochs using Adam optimizer, and its parameters were set to: $\alpha = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. A batch size of 500 samples was adopted, and dropout with a drop rate of 0.2 was used to prevent overfitting. The activation function used was the exponential linear unit (ELU) and linear function in the output layer. Table 4 fully details the train, validation, and test set errors. As observable, the test set errors are similar to those obtained on validation, with only a significant difference in the current-flow constraints on VCB and VCOTA. Moreover, its generalization ability is observed on FVC (tsmc65), with errors even lower than validation circuits. Figure 6 shows that the ANN is producing workable solutions.

Finally, to further demonstrate the value of this work, the proposed approach's performance was compared to a human expert designer. In this experiment, a human-designed placement for the VCOTA architecture was compared to an automatically generated placement using the approach here proposed. The ANN generated, for a single sizing, 500 different predictions, by shuffling the order of the devices in the input layer. Due to the batch capabilities of the model, the time required for this process is in the order of milliseconds (push-button speed). The best prediction was selected as the final solution produced

by the ANN. The two final placements are shown side-by-side in Figure 7. Table 5 compares the numeric values for the compactness, proximity, current flow, regularity, and boundary errors for both solutions. Since the post-processing fixes overlaps and symmetry errors, these are zero and are not compared. Still, no major importance is given to the compactness comparison, as the remaining common-mode feedback circuitry used to bias the original circuit (and fill the empty spaces in the layout) were not included in this work. Nonetheless, in every metric, the automatic approach improves upon the human-designed floorplan, presenting decreases of up to 15-fold in the case of the current-flow error. Overall, the automatic approach is capable of carefully balancing all the different constraints, resulting in a placement with around 70% improvement over the human-designed solution.

Table 2. Dataset details.

Topology	VCB [23]	FVC [24]		VCOTA [25]
Technology	umc130 nm	umc130 nm	tsmc65 nm	umc130 nm
Number of Devices	14		15	22
Number of Sizing Examples	5000	258	40	5979
Training Set	70%	70%	0%	70%
Validation Set	15%	15%	50%	15%
Number of Train and Validation Samples	4250	2213 *	1260 *	5082
Test Set/Number of Test Samples	15%/750	15%/391 *	50%/1260 *	15%/897

* Augmented data through shuffling and padding.

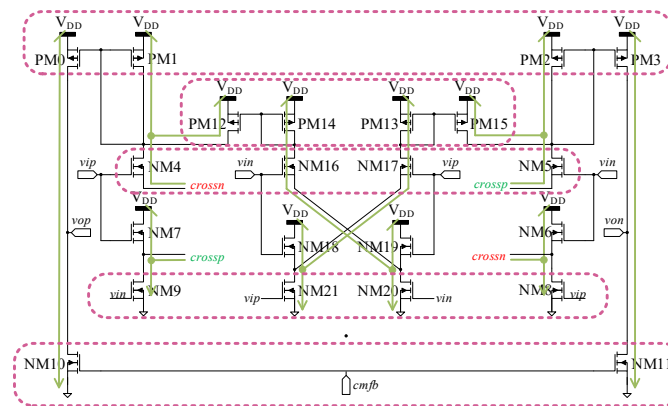


Figure 5. Schematic of the VCOTA [25]. Current-paths highlighted in green and row regularity constraints highlighted in pink.

Table 3. List of topological constraints specified for the VCOTA.

Symmetry			Current-Flow			Boundary			Row Regularity				Proximity		
PM0	PM3	PM0	→	NM10		Right	NM5	NM11	PM0	PM1	PM2	PM3	PM0	NM10	
PM1	PM2	PM3	→	NM11		Left	NM4	NM10	PM12	PM14	PM13	PM15	PM3	NM11	
PM12	PM15	PM1	→	NM4	→	Top	NM10	NM11	NM4	NM16	NM17	NM5	PM0	PM1	PM12
PM14	PM13	PM2	→	NM5	→				NM9	NM21	NM20	NM8	PM3	PM2	PM15
NM4	NM5	PM12	→	NM4	→				NM10	NM11			PM12	PM14	NM16
NM18	NM19	PM15	→	NM5	→								PM15	PM13	NM17
NM7	NM6	PM14	→	NM16	→								NM4	NM7	
NM17	NM16	PM13	→	NM17	→								NM5	NM6	
NM10	NM11	NM7	→	NM9									NM4	NM9	
NM9	NM8	NM6	→	NM8									NM5	NM8	
NM20	NM21	NM18	→	NM21									NM16	NM18	
		NM19	→	NM20									NM17	NM19	
													NM16	NM19	NM20
													NM17	NM18	NM21
													NM10	NM11	

Table 4. Train (TrE), validation (VaE), and test (TeE) errors.

Constraint	VCB [23]/umc130 nm				FVC [24]/umc130 nm				FVC [24]/tsmc65 nm				VCOTA [25]/umc130 nm			
	TrE	VaE	TeE	Comp. (%)	TrE	VaE	TeE	Comp. (%)	TrE	VaE	TeE	Comp. (%)	TrE	VaE	TeE	Comp. (%)
Wasted Area	0.90	0.86	0.85	-0.7	1.02	0.86	0.91	+5.2	-	0.67	0.59	-10.8	0.77	0.61	0.59	-3.6
Symmetry/SI ($\times 10^{-2}$)	0.62	9.47	9.18	-3.1	0.87	12.9	13.3	+3.1	-	10.6	9.75	-8.0	10.7	11.0	10.6	-3.6
Current-Flow ($\times 10^{-2}$)	0.14	2.13	2.60	+22.1	<0.1	1.78	1.90	+6.7	-	3.78	3.05	-19.3	<0.1	4.93	5.88	+19.2
Boundary ($\times 10^{-2}$)	0.92	2.71	2.91	+7.4	7.65	4.20	4.53	+7.9	-	3.89	2.11	-45.8	5.15	4.52	4.24	-6.2
Regularity ($\times 10^{-7}$)	3.01	1.27	1.32	+3.9	2.79	4.56	3.85	-15.6	-	2.84	2.78	-2.1	23.4	16.3	15.0	-8.0
Proximity	0.47	0.61	0.61	-0.8	0.21	0.43	0.44	+3.5	-	0.36	0.30	-17.9	0.21	0.34	0.32	-3.9
Overlap ($\times 10^{-2}$)	4.83	7.81	8.05	+3.1	6.13	12.7	12.1	-4.7	-	14.4	14.9	+3.5	7.85	15.1	15.4	+2.0

Table 5. Comparison of the automatically produced and human-designed floorplan solutions.

Placement	Compactness	Comp.	Proximity	Comp.	Current-Flow	Comp.	Regularity	Comp.	Boundary	Comp.	Total	Comp.
Human [25]	6.72	1.10×	8.33	2.19×	0.89	17.8×	0.54	4.50×	1.53	3.26×	18.01	1.70×
This Work	6.12	-	3.81	-	0.05	-	0.12	-	0.47	-	10.57	-

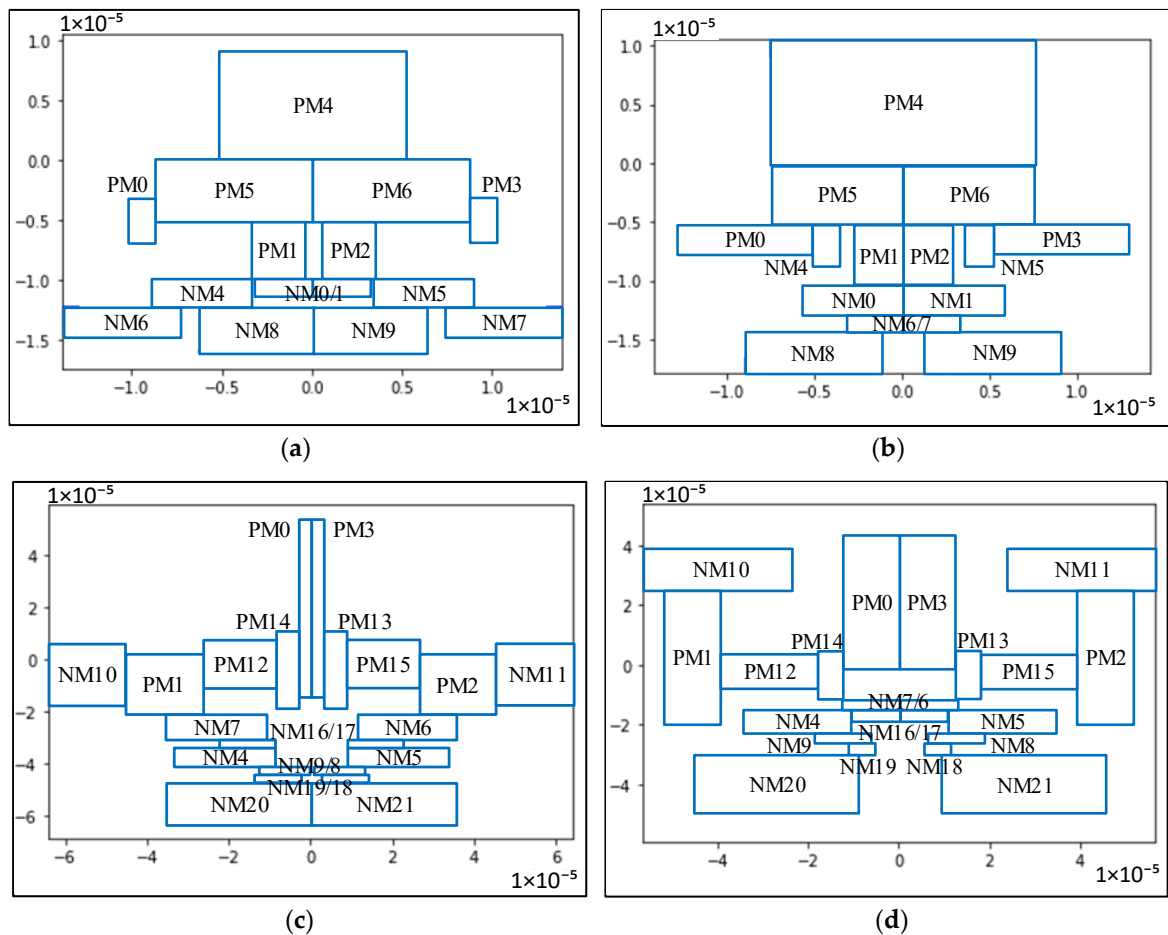


Figure 6. Test set predictions: (a,b) FVC (tsmc65); (c,d) VCOTA.

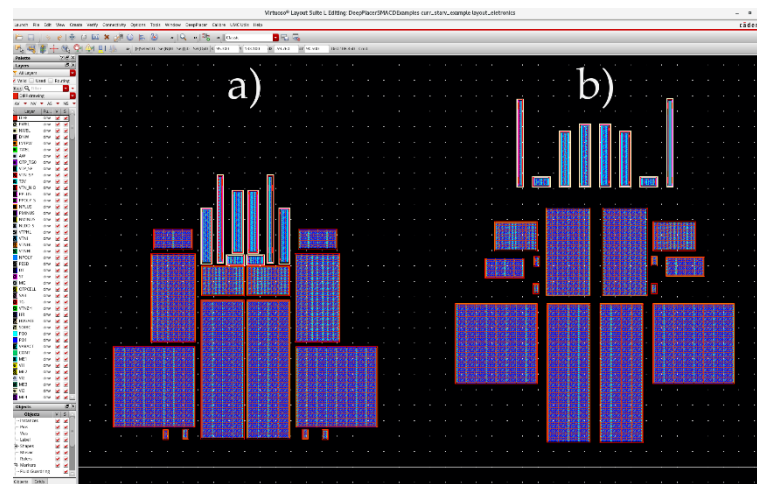


Figure 7. Placements generated for the VCOTA: (a) generated by this work, (b) designed by a human expert designer [25].

5. Discussion

Reliable automatic placement tools must support a broad set of topological constraints to tackle industry-standard analog IC design cases and produce aesthetic-wise solutions for the circuit designer. This paper proposes different model-independent and differentiable implementations of boundary, regularity, and SI constraints. The proposed constraints not only enforce the correct placement of the devices but were also designed to maximize the training of DL models. Their design was aware of the gradient produced and used matrix representation for improved performance using hardware parallelization (GPUs). The proof of concept was made with an ANN trained on sizing data of three different state-of-the-art amplifiers, generating workable solutions at push-button speed.

Even though the methodology proposed in this manuscript is tested in a family of state-of-the-art amplifiers, it can be applied to any real circuit class as long as there are well-defined topological constraints (i.e., symmetry pairs, current-flows, proximity, boundary, etc., constraints) for the circuit topologies within that class. If no topological constraints are requested, the deep models will still attempt to minimize the layout's wasted area/compactness and the overlap among devices, and components of the placement loss indicated in Equation (34). However, the resulting solution may not be meaningful for the circuit designer, as no criteria to place the devices with respect to each other is being set. The experiments carried intend to show that if the deep model is trained in a dataset that shares some of the characteristics regarding topological constraints' fulfillment of the circuit topologies to be generated, the acquired knowledge can be reused and generalized.

Even if the placement of a relatively small analog integrated circuit block is not extremely time-consuming, once several performance figures have to be considered its complexity increases immensely, and an optimal floorplan solution may not exist due to the inherent tradeoffs. Moreover, analog IC design is characterized by non-systematic re-design iterations, most of which impact the layout design, requiring partial or complete floorplan re-design. By using deep models, these floorplans can be generated instantly, while the human-designed floorplan may take from dozens of minutes to a couple of hours. These times are multiplied through every re-design iteration further conducted on a design.

Author Contributions: Conceptualization, A.G., N.L. and R.M.; methodology, A.G. and P.A.; software, A.G. and P.A.; validation, A.G. and P.A.; investigation, A.G. and P.A.; data curation, A.G. and P.A.; writing—original draft preparation, A.G. and P.A.; writing—review and editing, N.H., N.L. and R.M.; visualization, A.G. and N.L.; supervision, N.H., N.L. and R.M.; project administration, R.M.; funding acquisition, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/50008/2020 (including internal research project LAY(RF)²/X-0002-LX-20), and Research Grant FCT-SFRH/BD/07123/2021.

Data Availability Statement: Data available on request due to restrictions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Lin, P.H.; Chang, Y.W.; Lin, S.C. Analog Placement Based on Symmetry-Island Formulation. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2009**, *28*, 791–804. [\[CrossRef\]](#)
2. Patyal, A.; Chen, H.M.; Lin, M.P.H.; Fang, G.Q.; Chen, S.Y.H. Pole-Aware Analog Layout Synthesis Considering Monotonic Current Flows and Wire-Crossings. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2022**, *42*, 266–279. [\[CrossRef\]](#)
3. Wu, P.H.; Lin, M.P.H.; Chen, T.C.; Yeh, C.F.; Li, X.; Ho, T.Y. A Novel Analog Physical Synthesis Methodology Integrating Existent Design Expertise. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2015**, *34*, 199–212. [\[CrossRef\]](#)
4. Pan, P.C.; Chin, C.Y.; Chen, H.M.; Chen, T.C.; Lee, C.C.; Lin, J.C. A Fast Prototyping Framework for Analog Layout Migration with Planar Preservation. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2015**, *34*, 1373–1386. [\[CrossRef\]](#)
5. Han, J.; Bae, W.; Chang, E.; Wang, Z.; Nikolic, B.; Alon, E. LAYGO: A Template-and-Grid-Based Layout Generation Engine for Advanced CMOS Technologies. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 1012–1022. [\[CrossRef\]](#)
6. Unutulmaz, A.; Dündar, G.; Fernández, F.V. Template Coding with LDS and Applications of LDS in EDA. *Analog. Integr. Circuits Signal Process.* **2014**, *78*, 137–151. [\[CrossRef\]](#)
7. Afacan, E.; Lourenço, N.; Martins, R.; Dündar, G. Review: Machine Learning Techniques in Analog/RF Integrated Circuit Design, Synthesis, Layout, and Test. *Integration* **2021**, *77*, 113–130. [\[CrossRef\]](#)
8. Mina, R.; Jabbour, C.; Sakr, G.E. A Review of Machine Learning Techniques in Analog Integrated Circuit Design Automation. *Electronics* **2022**, *11*, 435. [\[CrossRef\]](#)
9. Gusmão, A.; Passos, F.; Póvoa, R.; Horta, N.; Lourenço, N.; Martins, R. Semi-Supervised Artificial Neural Networks towards Analog IC Placement Recommender. In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020. [\[CrossRef\]](#)
10. Guerra, D.; Canelas, A.; Póvoa, R.; Horta, N.; Lourenço, N.; Martins, R. Artificial Neural Networks as an Alternative for Automatic Analog IC Placement. In Proceedings of the SMACD 2019—16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Lausanne, Switzerland, 15–18 July 2019. [\[CrossRef\]](#)
11. Gusmão, A.; Póvoa, R.; Horta, N.; Lourenço, N.; Martins, R. DeepPlacer: A Custom Integrated OpAmp Placement Tool Using Deep Models. *Appl. Soft Comput.* **2022**, *115*, 108188. [\[CrossRef\]](#)
12. Gusmão, A.; Horta, N.; Lourenço, N.; Martins, R. Scalable and Order Invariant Analog Integrated Circuit Placement with Attention-Based Graph-to-Sequence Deep Models. *Expert Syst. Appl.* **2022**, *207*, 117954. [\[CrossRef\]](#)
13. Wu, I.P.; Ou, H.C.; Chang, Y.W. QB-Trees: Towards an Optimal Topological Representation and Its Applications to Analog Layout Designs. In Proceedings of the Design Automation Conference, Austin, TX, USA, 5–9 June 2016; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2016.
14. Sanabria-Borbón, A.C.; Soto-Aguilar, S.; Estrada-López, J.J.; Allaire, D.; Sánchez-Sinencio, E. Gaussian-Process-Based Surrogate for Optimization-Aided and Process-Variations-Aware Analog Circuit Design. *Electronics* **2020**, *9*, 685. [\[CrossRef\]](#)
15. Valencia-Ponce, M.A.; Tlelo-Cuautle, E.; de la Fraga, L.G. On the Sizing of Cmos Operational Amplifiers by Applying Many-Objective Optimization Algorithms. *Electronics* **2021**, *10*, 3148. [\[CrossRef\]](#)
16. Kunal, K.; Dhar, T.; Madhusudan, M.; Poojary, J.; Sharma, A.; Xu, W.; Burns, S.M.; Hu, J.; Harjani, R.; Sapatnekar, S.S. GANA: Graph Convolutional Network Based Automated Netlist Annotation for Analog Circuits. In Proceedings of the 2020 Design, Automation and Test in Europe Conference and Exhibition, Grenoble, France, 9–13 March 2020; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2020; pp. 55–60.
17. Xu, B.; Lin, Y.; Tang, X.; Li, S.; Shen, L.; Sun, N.; Pan, D.Z. WellGAN: Generative-Adversarial-Network-Guided Well Generation for Analog/Mixed-Signal Circuit Layout. In Proceedings of the 56th Design Automation Conference, Las Vegas, NV, USA, 2–6 June 2019; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2019.
18. Zhu, K.; Liu, M.; Lin, Y.; Xu, B.; Li, S.; Tang, X.; Sun, N.; Pan, D.Z. GeniusRoute: A New Analog Routing Paradigm Using Generative Neural Network Guidance. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Westminster, CO, USA, 4–7 November 2019. [\[CrossRef\]](#)
19. Eick, M.; Strasser, M.; Lu, K.; Schlichtmann, U.; Graeb, H.E. Comprehensive Generation of Hierarchical Placement Rules for Analog Integrated Circuits. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2011**, *30*, 180–193. [\[CrossRef\]](#)
20. Eick, M.; Graeb, H.E. MARS: Matching-Driven Analog Sizing. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* **2012**, *31*, 1145–1158. [\[CrossRef\]](#)

21. Liu, M.; Li, W.; Zhu, K.; Xu, B.; Lin, Y.; Shen, L.; Tang, X.; Sun, N.; Pan, D.Z. S3DET: Detecting System Symmetry Constraints for Analog Circuits with Graph Similarity. In Proceedings of the 25th Asia and South Pacific Design Automation Conference (ASP-DAC), Beijing, China, 13–16 January 2020; pp. 193–198. [[CrossRef](#)]
22. Kunal, K.; Poojary, J.; Dhar, T.; Madhusudan, M.; Harjani, R.; Sapatnekar, S.S. A General Approach for Identifying Hierarchical Symmetry Constraints for Analog Circuit Layout. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, Virtual Event, 2–5 November 2020. [[CrossRef](#)]
23. Pova, R.; Lourenco, N.; Martins, R.; Canelas, A.; Horta, N.C.G.; Goes, J. Single-Stage Amplifier Biased by Voltage Combiners with Gain and Energy-Efficiency Enhancement. *IEEE Trans. Circuits Syst. II Express Briefs* **2018**, *65*, 266–270. [[CrossRef](#)]
24. Pova, R.; Lourenco, N.; Martins, R.; Canelas, A.; Horta, N.; Goes, J. A Folded Voltage-Combiners Biased Amplifier for Low Voltage and High Energy-Efficiency Applications. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 230–234. [[CrossRef](#)]
25. Pova, R.; Lourenco, N.; Martins, R.; Canelas, A.; Horta, N.; Goes, J. Single-Stage OTA Biased by Voltage-Combiners with Enhanced Performance Using Current Starving. *IEEE Trans. Circuits Syst. II Express Briefs* **2018**, *65*, 1599–1603. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.