# Towards Decrypting the Art of Analog Layout: Placement Quality Prediction via Transfer Learning

Mingjie Liu*, Keren Zhu*, Jiaqi Gu, Linxiao Shen, Xiyuan Tang, Nan Sun, and David Z. Pan
ECE Department, The University of Texas at Austin, Austin, TX, USA
{jay_liu, keren.zhu, jqgu, lynn.shenlx, xitang}@utexas.edu, nansun@mail.utexas.edu, dpan@ece.utexas.edu

***Abstract*—Despite tremendous efforts in analog layout automation, little adoption has been demonstrated in practical design flows. Traditional analog layout synthesis tools use various heuristic constraints to prune the design space to ensure post layout performance. However, these approaches provide limited guarantee and poor generalizability due to a lack of model mapping layout properties to circuit performance. In this paper, we attempt to shorten the gap in post layout performance modeling for analog circuits with a quantitative statistical approach. We leverage a state-of-the-art automatic analog layout tool and industry-level simulator to generate labeled training data in an automated manner. We propose a 3D convolutional neural network (CNN) model to predict the relative placement quality using well-crafted placement features. To achieve data-efficiency for practical usage, we further propose a transfer learning scheme that greatly reduces the amount of data needed. Our model would enable early pruning and efficient design explorations for practical layout design flows. Experimental results demonstrate the effectiveness and generalizability of our method across different operational transconductance amplifier (OTA) designs.**

## I. Introduction

Analog and mixed-signal (AMS) integrated circuits (ICs) are widely used in many emerging applications, including consumer electronics, automotive, and Internet of Things. The increasing demand of these applications calls for a shorter design cycle and time-to-market. As technologies continue to scale, sensitive layout-dependent effects make robust and high-performance analog layout design a increasing challenge [1].

Implementing analog circuit layouts is a heavily manual, time-consuming and error-prone task. Human layout designers draw the layouts following layout conventions learned from experience. Although those layout conventions provide good guidelines, they cannot directly ensure or optimize for the post layout performance. Time-consuming iterative trial and error is often required for design closure. On the other hand, the endeavor to automate analog layout generation has been continuing for decades [2]. However, little adoption has been demonstrated in practical analog design flows. This is partially due to the unavailability of the tools and, more importantly, the limited guarantee towards performance the tools provide.

Traditional analog layout synthesis tools rely on various heuristic constraints rather than explicit optimization over the post layout performance. Heuristic constraints are based on human layout techniques and enforced during placement and routing. However, heuristic constraints-based methods are questionable in practical designs flows; hand-crafted constraints are often questionable in explainability and confidence, and lack flexibility and generality in meeting the detailed needs of different scenarios. Without a direct model to predict the post-layout performance, analog layout has been more considered as an art that stubbornly defies all attempts for automation [3].

Previous works on analog performance modeling were targeted towards modeling the effect of device parameters on the circuit performance [4]–[6]. Most of the proposed modeling techniques are pre-layout; i.e., they model the analog design behavior at the schematic level. Prior works targeted towards modeling performance at the layout level derive equations to analyze various layout effects. Lampaert et al. [7] use sensitivity analysis on circuit performance and estimate interconnect parasitic and mismatch effects. Ou et al. [8] adopt a similar methodology in mitigating layout dependent effects for improved circuit robustness. However, with increased device scaling, analytical sensitivity estimates of parasitics and mismatch over performance are no longer accurate. Therefore, a modeling framework that establishes a mapping between a given layout and the expected post layout performance is imperative yet still challenging.

On the other hand, recent studies have demonstrated success in applying machine learning techniques to model abstract layout guidelines and guide automatic physical design. The work of [9] uses a CNN model to predict the post-routing results from early macro placement solutions and [10] predicts routability from standard cell placements. Similar success is also shown in analog layout automation in applying generative machine learning models to analog routing [11] and well generation [12]. While works in digital domain often leverage automatic layout flows to generate high-quality training data, the works of [11], [12] rely on manual analog layouts for training. The difficulty in obtaining human layouts might explain the lack of research in modeling analog layout quality.

In this paper, we propose a new methodology to shorten the gap in post layout performance modeling for analog circuits. We propose to predict the relative layout quality with certain post layout performance as a quantitative metric, given a certain design space. Successful quality prediction will enable early design pruning for fast design space explorations. Our proposed approach leverages an automatic flow to generate high-quality labeled training data and a convolutional neural network (CNN) along with well-crafted placement features to

* Both authors contributed equally to this work.

predict the post layout performance. To overcome the high cost of obtaining labeled training data, we further propose a transfer learning scheme that reduces the amount of data needed under different design setups. Our main contributions are summarized as follows:

- To the best of the authors' knowledge, we are the first to propose a placement quality prediction model for fast design space explorations.
- We propose a method of automatically generating simulated layout training data and extracting effective placement features related to placement quality.
- We apply coordinate channels with 3D convolution layers for improved model performance.
- We propose a transfer learning scheme that significantly reduces the amount of data needed during training, while still achieving effective design pruning.

The remainder of this paper is organized as follows. Section II gives the background on design space pruning and formulates the quality prediction task; Section III explains in details our proposed method of extracting placement features and 3D CNN model; Section IV demonstrates the experimental results on transfer learning; Section V concludes the paper.

## II. Background and Preliminaries

In this section we first give a background on design space pruning in analog layout synthesis flows in Sec. II-A. We then formulate our placement quality prediction task in Sec. II-B.

### A. Design Space Pruning in Analog Layout Synthesis

To satisfy post layout performance requirements and achieve design closure, a feedback loop from post layout performance is needed in the development of practical layout synthesis flow. The work of [13] proposes a framework where the circuit is resized if the post layout performance metrics is not met. We suppose that the circuit is already well-designed and limit our scope exclusively to exploring the design space of layout implementations. Design exploration would thus actively search for satisfactory layout implementations in the design space based on the feedback results from simulations. Previous works of performance driven analog layout synthesis tools attempt to reduce the design space by analytically embedding the layout impact on performance into a cost function. However, with increased device scaling and complexity of layout dependent effects, these methods are no longer accurate.

### B. Motivation and Problem Formulation

One major bottleneck of design exploration is the runtime involved with post layout simulations. To obtain the final simulation results, the generated layout design after place and route need to be extracted for parasitics and then evaluated with transistor level simulations. The runtime of simulations normally dominates a single iteration cycle. An effective prediction on the post layout performance quality in the early steps of the design cycle would thus prune the design space and allow faster explorations.

With the motivation to prune design space, we propose to predict the layout quality with intermediate results after the placement step. We define the layout quality as the relative post layout performance with the performance distribution in some give design space. Our task if formulated into the following classification problem: Given the circuit and intermediate placement solutions of device boundary box, placement coordinates and pin locations, predict whether to prune the design in regards to the performance quality. In other words, the objective is to discard placement candidates that are expected to give poor post layout performance results in the early design stage.

## III. Placement Quality Prediction

In this section, we explain in details our method of predicting placement quality. We first explain the method of generating and labeling data with an automatic layout generator in Sec. III-A. The details of the extracted features from placement results are presented in Sec. III-B. We then introduce embedding coordinate channels for coordinate related machine learning tasks in Sec. III-C and our 3D CNN model in Sec. III-D. Finally we give a brief overview of our transfer learning scheme in Sec. III-E.

### A. Data Generation and Labeling

We generate layouts for training and testing with an automatic layout generator [14]. We enforce symmetry constraints in placement and routing for all layouts generated. The symmetry constraints and critical nets are provided by the circuit designer. An analytical placement engine is used with the objective of minimizing the following cost:

$$f_{Cost} = f_{WL} + a \cdot f_{BND}. \tag{1}$$

where $f_{WL}$ is the total weighted half-perimeter wirelength (HPWL) and $f_{BND}$ is the penalty term for violating a desirable boundary for the design based on aspect ratio and white space ratio. While generating different layouts of the same design, we keep $f_{BND}$ fixed and change the net weighting in $f_{WL}$. We generate different net weights by selecting different combinations of critical nets for a higher weight value.

Since the performance metric and layout design space differ significantly for different analog functional building blocks, we limit our study to operational transconductance amplifier (OTA) designs. The statistics and performance metric evaluated are shown in Table I. OTA1 and OTA2 have the same circuit schematic but different sizing. All generated layouts are LVS (Layout Versus Schematic) clean.

TABLE I: Data Statistics

| Design | Stage | Compensation | Layouts | Metric |
|---|---|---|---|---|
| OTA1 | 3 | Nested Miller | 16,376 | Offset |
| OTA2 | 3 | Nested Miller | 16,381 | Offset |
| OTA3 | 2 | Miller | 16,384 | Offset |
| OTA4 | 2 | None | 16,363 | CMRR |

After obtaining the post layout simulation results, we label the quality for each data point based on the relative rank

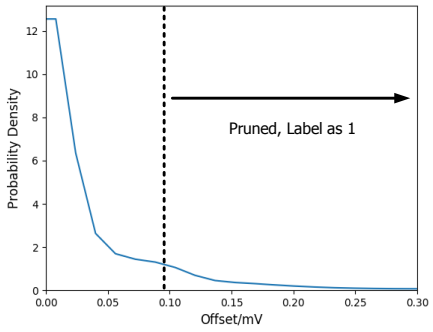
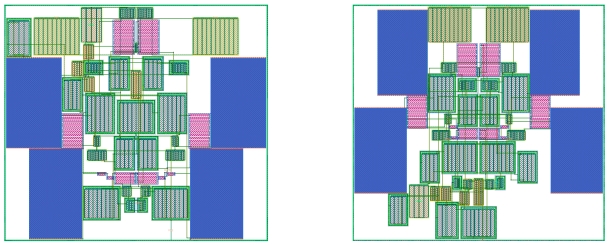



Fig. 1: Offset distribution of OTA1.



(a) (b)

Fig. 2: Example of layouts and labels. (a) Largest offset in OTA1, labeled 1 (b) Smallest offset in OTA1, labeled 0.

on performance. Figure 1 shows the distribution of the input referred offset (absolute value) of OTA1. A layout is labeled as being pruned if the performance is in the worst 25th percentile of the entire data set distribution. Figure 2 shows the worst and best layout of OTA1 and their corresponding labels.

### *B. Placement Feature Extraction*

The complex and intricate nature of analog circuit behaviors make extracting performance relevant features from placement extremely important. The performance impact of a device placement lies in both the placement location and circuit topology. As an example, the mismatch of differential input pairs has a larger impact towards offset compared with the load. Thus, to ensure a good and generalized model, extracted features have to be both easily extendable to different circuit topologies and able to encode effective placement information.

To leverage the success of convolutional neural networks in computer vision tasks, we represent intermediate layout placement results into 2D images. Instead of compacting the entire circuit placement into a single image, we separate devices into different images based on the circuit topology. For OTA circuits, we propose divide the circuit into the following subcircuits based on functionality:

- **First Stage** Devices in the first stage. This include the differential input, load, and tail transistors.
- **Other Stages** Devices in the other amplifier stages.
- **Feedback** Passive devices in the compensation feedback loop, such as miller capacitance.
- **CMFB** Common-mode feedback circuits.
- **Bias** Device in the current mirrors.
- **Routing Demand** The aggregated pin boundary box for each net.

Figure 3 shows the subcircuits of OTA3.

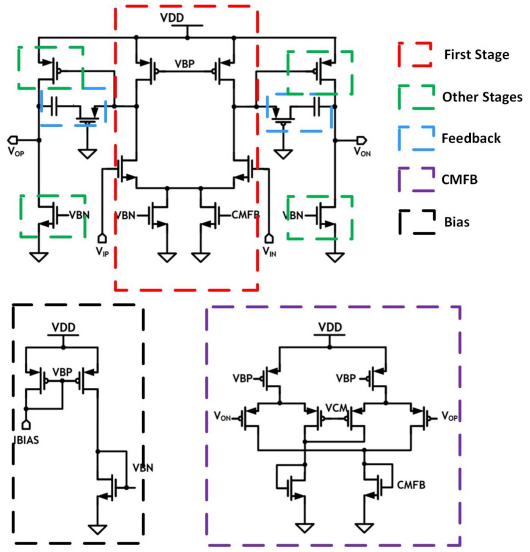



Fig. 3: Subcircuits of OTA3.

The devices are abstracted into rectangles and scaled according to the placement results into a image. In all our experiment, the image size is selected to be 64*64. We further encode different image intensities for device types shown in Table II. Figure 4 shows the corresponding extracted placement feature images of the layout in Fig 2(b).

TABLE II: Device Image Intensity

| Device | NMOS | PMOS | Capacitor | Resistor |
|---|---|---|---|---|
| Intensity | 0.25 | 0.5 | 0.75 | 1.0 |

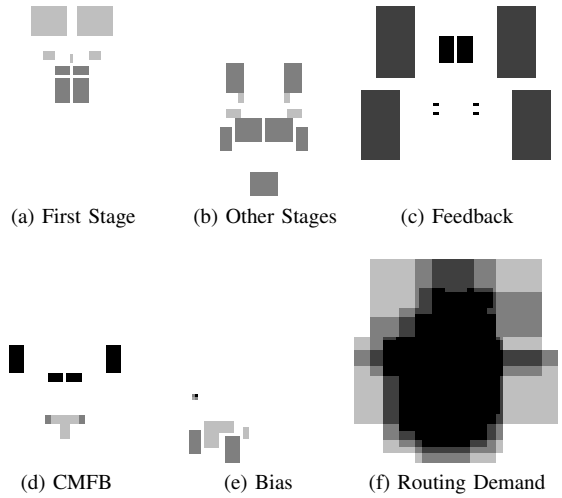
### *C. Embedding Coordinate Channels*

Traditional CNNs have been demonstrated ineffective at learning a mapping between coordinates in the Cartesian and image pixel space. Liu et al. [15] directly embed coordinate information through the use of extra channels, which greatly improved the model performance on location sensitive tasks such as object detection.

Since the placement quality will be directly affected by the distance between matching devices, we adopt a similar solution by adding extra coordinate channels to the feature images extracted in Sec. III-B. Algorithm 1 shows the method of embedding location features into extra coordinate channels.

### *D. 3D Convolutional Neural Networks*

Convolutional neural networks have been primarily applied on 2D images as a class of deep models for feature construction. Conventional 2D CNNs extract features from local neighborhoods on feature maps in the previous layer. Formally,



(a) First Stage (b) Other Stages (c) Feedback

(d) CMFB (e) Bias (f) Routing Demand

Fig. 4: Extracted feature images of Fig. 2(b). Device types are encoded in intensity.

**Algorithm 1** Adding Coordinate Channels

**Input:** Extracted feature image $Img$
**Output:** Additional coordinate channels $Coord_{x,y}$

```
function AddCoords(Img)
    Initialize Coord_{x,y} to 0
    for pixel (i, j) in Img do
        if Img(i, j) > 0 then
            Coord_x(i, j) ← i/dim_x
            Coord_y(i, j) ← j/dim_y
    return Coord_{x,y}
```

given the pixel value at position $(x, y)$ in the $j$th feature map in the $i$th layer, the convolutional layer output $v_{ij}^{xy}$ is given by

$$v_{ij}^{xy} = \sigma(\sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)} + b_{ij}), \quad (2)$$

where $\sigma(\cdot)$ is the activation function, $b_{ij}$ is the bias for feature map, $m$ indexes over the set of feature maps in this layer, $w_{ijm}^{pq}$ is the value of the weight kernel at the position $(p, q)$ connected to the $k$th feature map. The output feature is thus the activation output of a weighted sum over all the kernel maps with the previous layer images.

3D convolution layers were first proposed to incorporate both spacial and temporal information for action recognition in videos. In contrast to 2D CNNs where the convolution kernel is a 2D map, 3D convolution is achieved by convolving a 3D kernel to the cube formed by stacking multiple contiguous images together:

$$v_{ij}^{xyz} = \sigma(\sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)} + b_{ij}), \quad (3)$$

with $r$ being the value across the third dimension. Images captured across time from videos were stacked to form a 3D input tensor for action recognition [16]. The works of [17], [18] further demonstrated the effectiveness of 3D CNNs on capturing features of spacial 3D volumetric data.

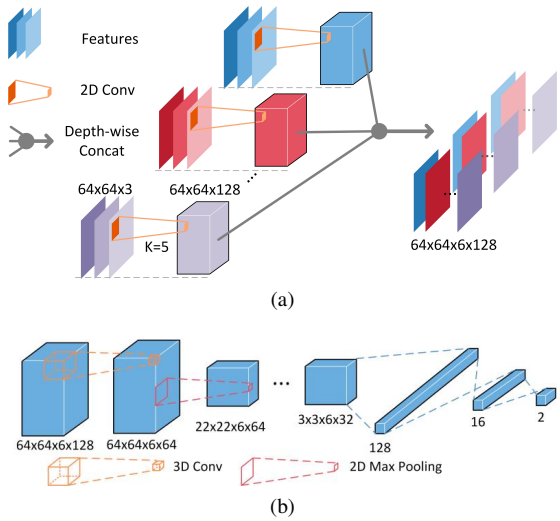



Fig. 5: Neural network architecture. (a) Initial separate 2D CNN. (b) 3D CNN classifier.

We propose the use of 3D CNNs to effectively capture the relative location information between the different placement subcircuits. Figure 5 shows the overall model of the 3D CNN network for placement quality prediction. Each extracted placement feature image is augmented into feature sets with coordinate channels as described in Sec.III-C. Initial features are then extracted separately for each feature sets with 2D convolutional layers. The outputs are then stacked to form 3D tensors. The 3D tensors are fed to the 3D CNN for placement quality prediction.

### *E. Transfer Learning*

Transfer learning represents a set of techniques to transfer the knowledge learned from source domain to a target domain [19]. In our setting, we hope to transfer a learned model with all the data from one design to predict the layout quality of another circuit design. We assume that the although the design space of different OTA circuits could be different, there are placement features related to the layout quality that could be shared between the source and target domains.

Our transfer learning scheme is in the inductive transfer learning setting, where labeled data are available in both the source and target domain. The model is first trained on the source domain where there is abundant labeled data. The pre-trained model is then finetuned with limited labeled data in the target domain. This method allows the model to preserve useful features learned from the source domain, and adapt to the specific task related to the target domain.

## IV. Experimental Results

We implemented the proposed placement feature extraction and 3D CNN model in Python. All layouts were generated in TSMC 40nm technology, extracted for parasitics with

Calibre PEX, and simulated with Cadence Spectre. For all our experiments, we select 20% of the data (around 3,200 layouts) to be the testing set which is never observed during training. The data set and feature extraction are open-source[1].

### A. Evaluation Metrics

For our application setting, we use the false omission rate $FOR$ as the key performance metric:

$$FOR = \frac{FN}{TN + FN}, \tag{4}$$

where $FN$ is the number of not pruned bad designs and $TN$ are the designs correctly selected to explore. False omission rate ($FOR$) measures the leakage of bad designs to design exploration. Without any pruning, none of the bad designs would be filtered and we would have a $FOR$ of 25%. We also report the accuracy, precision, recall and $F_1$ score in our results. A good layout quality prediction would have high accuracy, high precision and low $FOR$.

### B. Baseline Model and CNN Architecture Comparison

We use a balanced labeled data of OTA1 for training the baseline model for transfer learning. Similar to the labeling method described in Sec.III-A, we create a balanced data with the worst performing 25th percentile layout labeled as 1 and the best 25th as 0. Intuitively, this is exposing the best and worst placements to the machine learning model.

We experiment with different neural network architectures. *nofeat* indicate compacting the placement result into a single image, while *feat* is separating different subcircuits into multiple images and embedding extra coordinate channels. *3D* is the proposed 3D neural network architecture while *2D* is replacing all the convolution filters with 2D. Table III compares the training and testing accuracy for different architecture models. The proposed feature extraction with 3D CNN achieves the best testing accuracy.

TABLE III: Baseline Model Comparisons

| Model | Training Accuracy | Testing Accuracy |
|---|---|---|
| *nofeat+2D* | 97.95% | 78.44% |
| *nofeat+3D* | 79.23% | 78.32% |
| *feat+2D* | 96.19% | 91.94% |
| ***feat+3D*** | 95.51% | **93.83**% |

### C. Transfer Learning with Limited Data

We experiment on the transfer learning scheme proposed in Sec. III-E. For the transfer learning results, we report the evaluation metrics of the testing set after training with a reduced learning rate and compare with retraining.

Table IV reports the results on transfer learning. Training ratio $\alpha$ is defined as the the percentage of training data used in respect to the entire data set. Using the entire training set would have a training ratio of $0.80$ since the rest 20% is reserved for testing. A training ratio of $0.00$ indicate directly using the pre-trained baseline model without finetuning on any target domain data. We only report OTA1 with $\alpha = 0.80$ since the baseline is trained on this design. Based on the results, we make the following observations:

- Transfer learning significantly improves the results compared with retraining from random initialization.
- The performance of prediction and the effectiveness of pruning increases with the amount of training data.
- Even with limited training data of only 160 layouts, the placement quality prediction is quite effective.
- Directly applying the baseline model without finetuning is non-ideal, since the data distribution of the target and source domains could vary significantly.

With training on 10% of the data, our proposed transfer learning approach can achieve an average $FOR$ of 8.95% compared to 22.91% in the baseline setup. On OTA1, our method significantly reduces the $FOR$ by 57% compared with the baseline using only 1% of the data. The data efficiency demonstrated in turn results in a significant reduction in the exploration cost. With our model achieving up to 90% accuracy, we can prune more than 20% of the design space of low performance quality, while largely allowing designs of high quality to be explored.

### D. Transfer Learning with Few-shot Examples

In practical situations of design exploration, obtaining performance results of even a hundred layout might be expensive. Furthermore, the performance distribution could only be known until the design space has been fully explored. To further demonstrate the models effectiveness for early design pruning, we experiment transfer learning with only a few example from the target domain.

Our experiment setting is as follows. For every experiment, we randomly sample 16 layouts as the transfer training data. We label the training data according to their relative rank in the training set instead of the entire design space. We then relabel the testing set according to the critical value in the training distribution. The number of positive data in the testing set could vary significantly from 25%. The confidence of our model would thus be tested on the performance distribution of the training data instead of the entire design space. We repeat our experiment 100 times for each transfer target design.

Figure 6 shows the result on the few-shot transfer learning. The black line plots the false omission rate of random design pruning. With extremely limited data, the improvement gained with few-shot learning is highly correlated with the transfer task. Transferring the knowledge from the same design but different sizing (OTA2) is extremely effective. There is only limited improvement in few-shot transfer learning to a different design with different performance metric (OTA4).

## V. Conclusion

In this paper, we propose a new method of early layout design pruning by predicting the placement quality. Our 3D CNN model with well-crafted placement features offers enhanced flexibility, capable of generalizing to different OTA designs. We further propose a transfer learning scheme that

[1]https://github.com/magical-eda/UT-AnLay

TABLE IV: Transfer Learning Results

| Design | $\alpha$ | Accuracy (%) | | Precision (%) | | Recall (%) | | $F_1$ Score | | $FOR$ (%) | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | w | w/o | w | w/o | w | w/o | w | w/o | w | w/o |
| OTA1 | 0.80 | 90.29 | – | 86.69 | – | 73.79 | – | 0.7972 | – | 8.32 | – |
| OTA2 | 0.80 | 90.96 | 92.61 | 84.75 | 84.64 | 77.97 | 85.07 | 0.8122 | 0.8485 | 7.12 | **4.99** |
| | 0.10 | 90.10 | 80.40 | 82.53 | 68.07 | 76.78 | 17.74 | 0.7955 | 0.2815 | **7.57** | 22.00 |
| | 0.01 | 88.28 | 74.05 | 80.74 | 43.85 | 69.93 | 21.37 | 0.7495 | 0.2875 | **9.60** | 22.38 |
| | 0.00 | 70.10 | – | 52.47 | – | 25.06 | – | 0.3392 | – | 21.28 | – |
| OTA3 | 0.80 | 90.23 | 91.33 | 86.51 | 84.76 | 73.61 | 80.95 | 0.7954 | 0.8281 | 8.38 | **6.26** |
| | 0.10 | 87.29 | 79.98 | 77.52 | 58.07 | 69.78 | 80.95 | 0.7345 | 0.1385 | **9.75** | 23.84 |
| | 0.01 | 81.21 | 77.32 | 64.79 | 57.46 | 58.78 | 17.44 | 0.6118 | 0.2676 | **13.40** | 26.76 |
| | 0.00 | 74.73 | – | 49.81 | – | 40.25 | – | 0.4452 | – | 18.72 | – |
| OTA4 | 0.80 | 89.81 | 92.05 | 77.87 | 81.69 | 82.15 | 88.06 | 0.7995 | 0.8476 | 6.06 | **4.09** |
| | 0.10 | 88.70 | 74.33 | 81.69 | 48.96 | 70.01 | 15.80 | 0.7540 | 0.2389 | **9.54** | 22.90 |
| | 0.01 | 81.05 | 76.68 | 68.87 | 56.50 | 42.71 | 20.13 | 0.5272 | 0.2968 | **16.95** | 21.92 |
| | 0.00 | 49.72 | – | 26.78 | – | 59.54 | – | 0.3694 | – | 22.77 | – |

*w* are the transfer learning results. *w/o* are the results trained from random initialized weights.

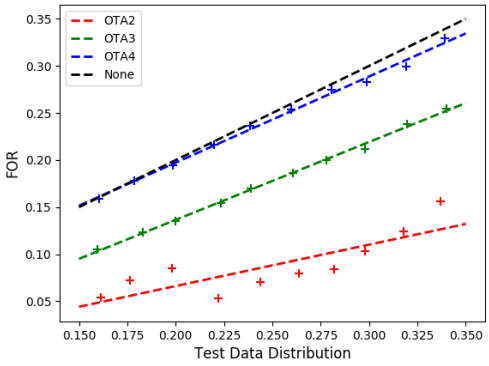



Fig. 6: Few-shot transfer learning results.

greatly reduces the amount of labeled data needed, achieving up to 57% reduction in the false omission rate compared to retraining the model, while using only 1% of labeled data. With our model, we can effectively prune more than 20% of the design space of low performance quality.

## ACKNOWLEDGEMENT

This work is supported in part by the NSF under Grant No. 1704758, and the DARPA ERI IDEA program. The authors would like to thank Mohamed Baker Alawieh from The University of Texas at Austin for helpful comments.

## REFERENCES

[1] R. A. Rutenbar, "Analog circuit and layout synthesis revisited," in *ISPD*, 2015, pp. 83–83.
[2] M. P.-H. Lin, Y.-W. Chang, and C.-M. Hung, "Recent research development and new challenges in analog layout synthesis," in *ASPDAC*, 2016.
[3] A. Hastings, *The Art of Analog Layout*. Prentice, 2005.
[4] F. Wang, P. Cachecho, W. Zhang, S. Sun, X. Li, R. Kanj, and C. Gu, "Bayesian model fusion: large-scale performance modeling of analog and mixed-signal circuits by reusing early-stage data," *IEEE TCAD*, vol. 35, no. 8, pp. 1255–1268, 2015.
[5] M. B. Alawieh, S. A. Williamson, and D. Z. Pan, "Rethinking sparsity in performance modeling for analog and mixed circuits using spike and slab models," in *DAC*, 2019.
[6] F. Gong, Y. Shi, H. Yu, and L. He, "Variability-aware parametric yield estimation for analog/mixed-signal circuits: Concepts, algorithms, and challenges," *IEEE Design & Test*, vol. 31, no. 4, pp. 6–15, 2014.
[7] K. Lampaert, G. Gielen, and W. M. Sansen, "A performance-driven placement tool for analog integrated circuits," *JSSC*, vol. 30, no. 7, pp. 773–780, 1995.
[8] H.-C. Ou, K.-H. Tseng, J.-Y. Liu, I. Wu, Y.-W. Chang *et al.*, "Layout-dependent-effects-aware analytical analog placement," in *DAC*, 2015.
[9] Y.-H. Huang, Z. Xie, G.-Q. Fang, T.-C. Yu, H. Ren, S.-Y. Fang, Y. Chen, and J. Hu, "Routability-driven macro placement with embedded cnn-based prediction model," in *DATE*, 2019.
[10] C. Yu and Z. Zhang, "Painting on placement, forecasting routing congestion using conditional generative adversarial nets," in *DAC*, 2019.
[11] K. Zhu, M. Liu, Y. Lin, B. Xu, S. Li, X. Tang, N. Sun, and D. Z. Pan, "Geniusroute: A new analog routing paradigm using generative neural network guidance," in *ICCAD*, 2019.
[12] B. Xu, Y. Lin, X. Tang, S. Li, L. Shen, N. Sun, and D. Z. Pan, "Wellgan: Generative-adversarial-network-guided well generation for analog/mixed-signal circuit layout," in *DAC*, 2019.
[13] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanović, "Late breaking results: Analog circuit generator based on deep neural network enhanced combinatorial optimization," in *DAC*, 2019.
[14] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan, "Magical: Toward fully automated analog ic layout leveraging human and machine intelligence," in *ICCAD*, 2019.
[15] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," in *Conference on Neural Information Processing Systems (NIPS)*, 2018.
[16] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, Jan 2013.
[17] Q. Dou, H. Chen, L. Yu, L. Zhao, J. Qin, D. Wang, V. C. Mok, L. Shi, and P. Heng, "Automatic detection of cerebral microbleeds from mr images via 3d convolutional neural networks," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1182–1195, May 2016.
[18] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[19] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.