Lithography Hotspot Detection using Deep Learning

1st Ajay Verma
Electrical Engineering
IIT Bombay
India
213070059@iitb.ac.in

2nd K Akhilesh Rao
Electrical Engineering
IIT Bombay
India
213079018@iitb.ac.in

3rd Dhruva S Hegde
Electrical Engineering
IIT Bombay
India
213079022@iitb.ac.in

***Abstract*—Lithography hotspot is a place on a VLSI layout which is susceptible to have an undesired open circuit or short circuit. This happens due to a mismatch between lithography wavelength and semiconductor technology feature size. Hotspot detection affects the turn-around time and the yield of IC manufacturing. Analytic methods used to detect hotspots involve many simulations and computations, which take a lot of time. Hence, machine learning approaches are being explored in order to quickly detect patterns that can have hotspots.**

***Index Terms*—Photo-lithography, Integrated Circuits, VLSI Layouts, Machine Learning**

I. Introduction

Photo-lithography is the most common method for semiconductor fabrication of ICs. It can create extremely small patterns, down to a few tens of nano-meters in size. It provides precise control of the shape and size of the objects it creates and can create patterns over an entire wafer in a single step, quickly and with relatively low cost.

As the size of integrated circuits (ICs) continues to shrink, the lithographic printability of the design becomes one of the important issues in IC design and manufacturing. The mismatch between lithography wavelength and semiconductor technology feature causes diffraction when transferring the layout from design onto a silicon wafer. This can cause hotspots in the printed layouts. Classic types of hotspots are-

- Pinching hotspots : Open Circuits
- Bridging hotspots : Short Circuits

Both of these are illustrated in Fig. 1.

To solve the problem of functional and parametric yields loss caused by these sensitive layout patterns, an efficient and accurate tool for lithographic hotspot detection at early design stages is in great demand.

Lithographic simulation offers an accurate prediction and detection of lithography hotspots. However, such simulations require heavy computation resources and run-time overhead. To speed up the hotspot detection process, pattern matching and machine learning approaches need to be explored.
Pattern matching relies only on the pre-defined pattern library and cannot detect unseen hotspot patterns, resulting in low prediction accuracy. Therefore, conventional machine learning and deep learning methods are viable options (Fig. 2).

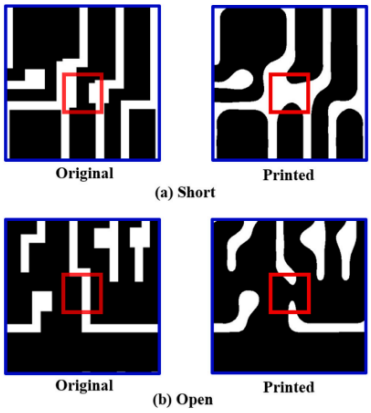

Fig. 1. Examples of types of hotspot patterns

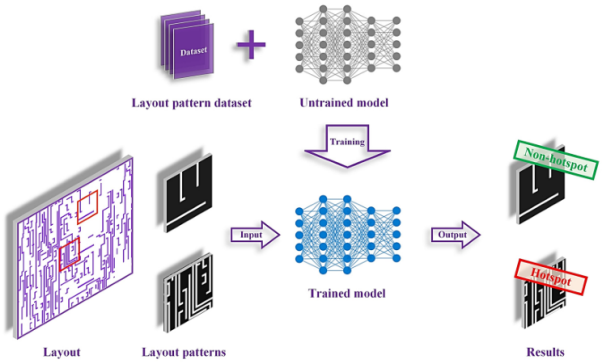

Fig. 2. Schematic diagram of hotspot detection using deep learning

II. Preliminaries

A. Data-set Analysis

International Conference on Computer-Aided Design (ICCAD) had released an open-source data-set in 2012 to evaluate hotspot detection methods, which has become the most widely used benchmark. This data-set is usually referred to as the ICCAD-12 benchmark data-set and it is used to train and test lithography hotspots in this work.

The ICCAD-12 data-set consists of 5 difference benchmarks. The following table (Fig. 3) lists the statistics of each of the benchmarks.

Benchmark	Training		Testing	
	HS#	NHS#	HS#	NHS#
ICCAD-1	99	340	226	3869
ICCAD-2	174	5285	498	41298
ICCAD-3	909	4643	1808	46333
ICCAD-4	95	4452	177	31890
ICCAD-5	26	2716	41	19327

Fig. 3. ICCAD-12 statistics

Note that all the benchmarks are highly imbalanced. This is because the number of patterns that can cause hotspots are relatively much less than the number of patterns that don't cause hotspots.

B. Model Selection

In this problem, both accuracy and inference time are important because it is essential to catch potential hotspots and do it quickly in order to reduce the time overhead. Therefore in this work, two different approaches have been taken and compared with respect to different metrics.

- The first approach is to use transfer learning on a pre-trained CNN model (VGGNET-16). Features extracted from VGGNET-16 are used to train classifiers. [2] [3]
- The second approach is to build and train a custom light-weight convolutional neural network (CNN) model from scratch. [1]

C. Evaluation Metrics

Due to the unbalanced nature of the data, plain accuracy is a misleading metric to measure the performance of a model. Therefore, entire confusion matrix (refer to Fig. 4) is visualized and metrics such as Precision, Recall, F1-score, Sensitivity, Specificity and Balanced Accuracy are calculated.

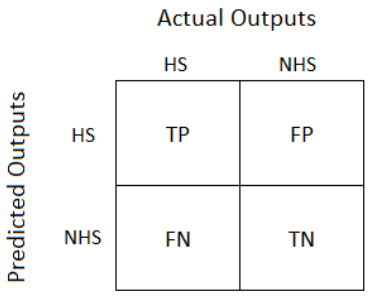

Fig. 4. Confusion Matrix

Formulae for the evaluation metrics:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{FP + TN}$$

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$Balanced\ Accuracy = \frac{Sensitivty + Specifity}{2}$$

In particular, Balanced Accuracy is the best metric to measure the performance of a model when the data is imbalanced.

III. Transfer Learning on VGGNET-16

VGGNET-16 is a popular CNN model that has been pre-trained on the ImageNet dataset. The entire architecture of VGGNET-16 is illustrated in Fig. 5.

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

Fig. 5. VGGNET-16 Summary

In order to use this model for our use-case, the last 2 fully connected layers have been removed. Hence, effectively 4096 features are extracted from the images of layouts.

The extracted features are then trained on the following 4 classifiers:

1) Support Vector Machine with Polynomial Kernel
2) Support Vector Machine with RBF Kernel
3) Random Forest
4) k-Nearest Neighbors

Performance of each of these classifiers on validation data is measured and the best classifier is chosen.

IV. Custom CNN

There are two draw-backs of using transfer learning for this use-case. First, since pre-trained models are trained on images of real world objects, it is unlikely that the features extracted are suitable for images of layouts. And second, the computational cost of using the entire CNN model such as VGGNET-16 is very high (it contains over 117,000,000 parameters) and hence it may not be suitable to integrate with CAD tools.

Keeping in mind the necessity of fast inference required to integrate hotspot detection into a CAD tool, the following light-weight CNN architecture is constructed (Refer to Fig. 6 and Fig. 7).

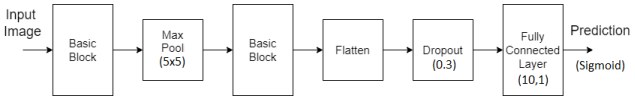

Fig. 6. Custom CNN Architecture

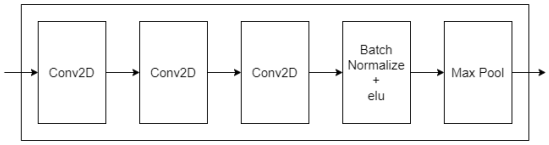

Fig. 7. Architecture of Basic Block

Hyper-parameters of the basic block are given in TABLE I. The total number of parameters in this model is 12,873. This is 4 orders of magnitude less than the VGGNET-16 model, making it very lightweight.
This model is trained using 'Nadam' optimizer for 5-10 epochs and balanced accuracy is measured.

Layer	Hyper-parameters
Conv2D	Filters=12, Window Size=(3,3), Activation=elu
Conv2D	Filters=12, Window Size=(3,3), Activation=elu
Conv2D	Filters=12, Window Size=(3,3), Activation=None
Batch Normalize	momentum=0.99, epsilon=0.001
Activation	elu
Max Pool	Window Size = (2,2)

TABLE I
HYPER-PARAMETERS OF A BASIC BLOCK

V. RESULTS & CONCLUSIONS

As mentioned earlier, due to the high imbalance in the positive and negative samples in the datasets, balanced accuracy is used as the evaluation metric.

A. Transfer Learning

Fig. 8 shows the validation balanced accuracy of each of the 4 classifiers on the ICCAD-12 benchmark datasets.

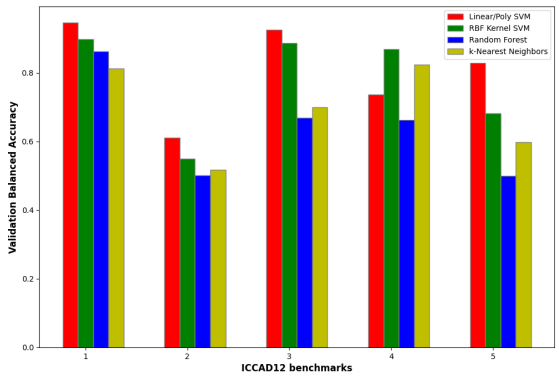

Fig. 8. Transfer Learning : Validation Balanced Accuracy of each classifier

Hotspot detection using transfer learning gives fairly good results. It can be seen that Linear SVM outperforms all other classifiers in 4 of the 5 benchmarks, with the maximum value being 94%. Typically about 80% balanced accuracy is achieved, with the exception of benchmark 2, on which the performance is particularly bad.

Model	Balanced Accuracy (in %)
Linear/Poly SVM	80.94
RBF Kernel SVM	77.68
Random Forest	63.88
k-Nearest Neighbors	68.98

TABLE II
AVERAGE BALANCED ACCURACY

Inference time of the best model i.e Linear SVM (trained on benchmark 1) is evaluated on a random dataset of 200 images. Average inference time per classification (on Colab GPU) is calculated to be 0.182 seconds (or 182 ms).

B. Custom Model

Fig. 9 shows the validation balanced accuracy of custom CNN on the ICCAD-12 benchmark datasets.

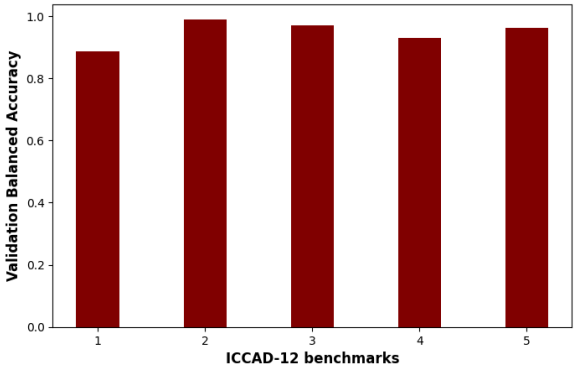

Fig. 9. Custom CNN : Validation Balanced Accuracy

The custom model performs better even though it is more lightweight, which may be counter-intuitive. But because it is trained from scratch, the CNN is able to extract relevant features.

Refer to Fig. 10 to see the comparison between the best model obtained by transfer learning and the custom model. Apart from benchmark 1 (where the performances are equal), the custom model outperforms the best transfer learning model. The best performance is 97.9%, which (ironically) is obtained on benchmark 2. The average balanced accuracy is 95.3%.

Inference time of custom CNN model (trained on benchmark 1) is evaluated on a random dataset of 200 images. The average inference time per classification is measured to be 0.0063 seconds (or 6.3 ms). This achieves 28.8 times speed-up compared to transfer learning model.

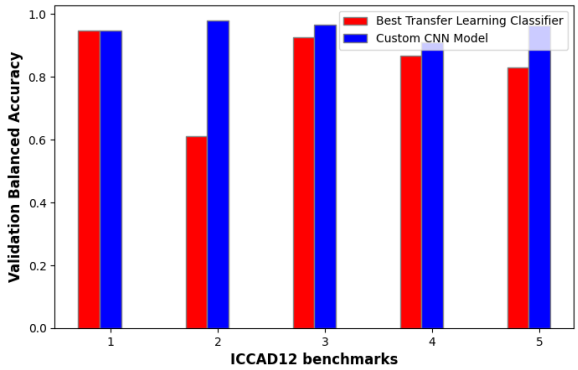

Fig. 10. Comparison of Validation Balanced Accuracy between Transfer Learning Model and Custom CNN Model

Therefore, it can be concluded that custom CNN model performs better than transfer learning model in both accuracy as well as inference time, making it more suitable to integrate with CAD tools.

REFERENCES

[1] V. Borisov and J. Scheible, "Lithography Hotspots Detection Using Deep Learning," 2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Prague, Czech Republic, 2018, pp. 145-148, doi: 10.1109/SMACD.2018.8434561.

[2] Liao, L.; Li, S.; Che, Y.; Shi, W.; Wang, X. Lithography Hotspot Detection Method Based on Transfer Learning Using Pre-Trained Deep Convolutional Neural Network. Appl. Sci. 2022, 12, 2192. https://doi.org/10.3390/app12042192

[3] H. Yang, Y. Lin, B. Yu and E. F. Y. Young, "Lithography hotspot detection: From shallow to deep learning," 2017 30th IEEE International System-on-Chip Conference (SOCC), Munich, Germany, 2017, pp. 233-238, doi: 10.1109/SOCC.2017.8226047.

[4] https://www.tensorflow.org/

[5] https://keras.io/api/applications/

[6] https://scikit-learn.org/stable/