A Systematic Study of Hotspot Detection in Physical Designs using Machine Learning

Piyush Verma[1], Robert Pack[2] and Sriram Madhavan[2]
[1]GLOBALFOUNDRIES, 400 Stonebreak Road Extension, Malta, NY, USA 12020
[2]GLOBALFOUNDRIES Inc, 2600 Great America Way, Santa Clara, CA, USA 95054

Abstract— **At advanced VLSI nodes, detecting potential yield detractors in the physical design is becoming increasing challenging. These design weak points or hotspots tend to be complex geometric patterns that pass all the design rules but are very difficult to manufacture. In this article, we demonstrate how machine learning based techniques can be used to detect these hotspots in a VLSI design. We propose a scalable data generation flow that can be used to train any machine learning model. We use this flow to generate a large balanced dataset and train several models to systematically study the effects of various parameters like the dataset size, the clip diameter and the number of extracted features. We test several standard machine learning algorithms with this dataset and finally demonstrate models with very high hotspot detection accuracy.**

I. Introduction

As VLSI technology continues to evolve, the manufacturing process is being pushed towards lower-k1 sub-wavelength printing techniques and structurally intricate integration schemes. Consequently, physical design verification from the perspective of manufacturability is becoming increasingly challenging. Complex geometric patterns that pass all the design rules, end up limiting the silicon yield of the design. These patterns are referred to as design hotspots. Early detection of these hotspots can result in tremendous cost savings over the lifetime of the product.

Currently, the hotspots are detected in the mask tapeout flow by using process simulations. However, despite numerous advancements in these simulators, an accurate and fast full chip 3D simulator that accurately represents all the process steps in the manufacturing flow is lacking. Moreover, existing physics based single step process simulators are slow and expensive to calibrate [1], [2], [3], [4].

Machine learning based techniques are suitable for problems where a large volume of data is available and a large number of parameters contribute to the final outcome. Hotspot detection fits this criteria very well. Recently, several groups have demonstrated the application of machine learning to lithography hotspot prediction in VLSI designs [5], [6], [7], [8], [9], [10], [11], [12]. These techniques classify clips from a large VLSI design as either a hotspot or a non-hotspot. While these studies show good results, they lack a comprehensive evaluation of various parameters like the dataset size, hotspot diameter etc and the choice of machine learning algorithm. Typically, the model is trained using a dataset consisting of a few hundred to a few thousand datapoints with the objective of identifying the presence of a hotspot close to the center of the clip. This is not ideal for a practical verification flow since a random clip from a large physical design might hide a hotspot anywhere within its extent. A model trained to predict the presence of a hotspot at the center of a clip will misclassify the clip and result in poor prediction performance.

We propose a labeled data generation flow which can easily be scaled to very large datasets without running into the issue of class imbalance [13]. The dataset generated using this flow can be used to train a machine learning model and classify a given layout clip as either having a hotspot or not having one as shown in figure 1. The hotspot does not need to be at the center of the clip for the it to be classified as a hotspot clip. To understand the underlying trade-offs between model prediction accuracy and the required computational resources, we present a comprehensive study of various machine learning algorithms against several parameters like the size of the training dataset, the size of hotspot clips and the number of features.

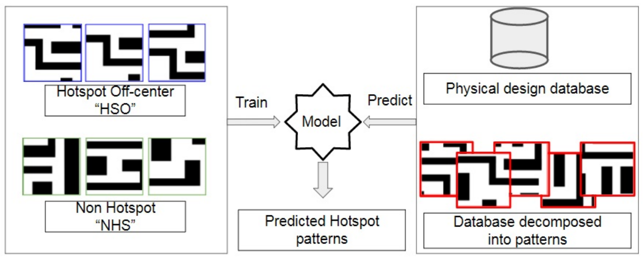

Fig. 1: Flowchart for detecting hotspots in a physical VLSI design.

II. Methodology

This section describes the method used to generate encoded data from physical designs, and the procedure followed to select, train and test various machine learning algorithms.

A. *Labeled data generation*

Labeled data generation involves the generation of feature vectors labeled as either hotspot or non-hotspot from the physical design and an associated list of weak point or hotspot locations. This labeled dataset is used to train and test a supervised machine learning model as described in a later section. For the experiments described in this article, a physical layout about 200 mm^2 in area with an associated hotspot site list containing 48,054 locations was used. The hotspots were all of the same type, pinching or line break which cause an open in a metal wire. Two different clip or pattern sizes were picked for all the experiments in this article based on photolithography considerations. A large clip diameter corresponding to 20 times the minimum pitch of the metal wire and a small clip diameter corresponding to 10 times the minimum pitch of the metal wire. To assess the pattern diversity across the hotspot locations on the design, an exact pattern classification was performed at the hotspot sites. 48,054 locations folded into 48,035 unique patterns at the large clip diameter and 47,997 unique patterns at the small clip diameter. Since the ratio of unique pattern count to total hotspot count is close to 1 for both clip diameters, the hotspot sites have a very high pattern diversity.

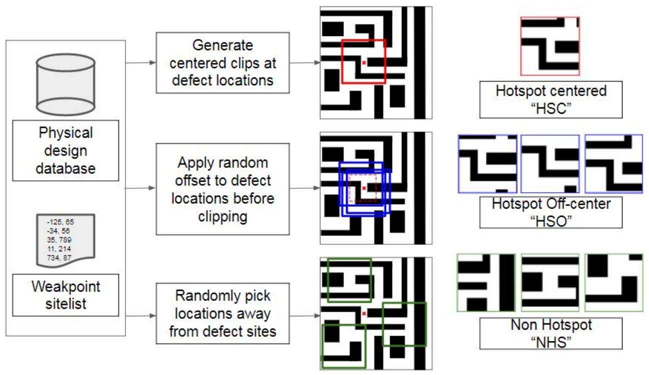
Physical design database
-125, 65
-34, 56
35, 789
11, 214
734, 87
Weakpoint sitelist
Generate centered clips at defect locations
Apply random offset to defect locations before clipping
Randomly pick locations away from defect sites
Hotspot centered "HSC"
Hotspot Off-center "HSO"
Non Hotspot "NHS"

Fig. 2: Flowchart for generating labeled data from a given physical design with known hotspot locations.

The data generation process consists of two steps. The first step generates a library of physical design clips or patterns belonging to the hotspot and non-hotspot classes as shown in figure 2. The hotspot clips are further divided into two different classes - HSC (Hotspot Centered) and HSO (Hotspot Off-center). The HSC clips are defined as those layout clips which have a known hotspot exactly at the center of the clip. On the other hand, the HSO clips are defined to be those clips where the known hotspot is off-centered, while still within the extent of the clip boundary.

The HSC (Hotspot Centered) clips are generated by clipping out a square layout snippet with a given diameter at each hotspot site present in the associated hotspot site list from the physical design. Since each hotspot site in the site list is used exactly once, the total number of HSC clips generated is equal to the number of hotspots in the site list. For the experiments described in this article, a total of 48,054 HSC clips were generated for each clip diameter as shown in table I.

The HSO (Hotspot Off-center) clips are generated by first applying a different random offset to each hotspot location in the hotspot site list and then clipping out a square snippet with a given diameter at these offset sites from the physical design. The maximum bound for the random offset is fixed at 75% of clip radius so that the hotspot is guaranteed to be contained within the central 75% region of the clip extent. By imposing this constraint on the random offset, the context

TABLE I: Table showing the count of clips collected for each of the 3 clip families for performing the machine learning experiments.

Pattern class	Total count (Large clip diameter)	Total count (Small clip diameter)
HSC	48,054	48,054
HSO	242,784	242, 625
NHS	257,216	257,375

around the hotspot is retained within the clip extent. This is an important consideration in the generation of HSO clips since the hotspot and the context around it are what truly define a hotspot. For the experiments described in this article, about 5 different HSO clips were generated per HSC clip, for each clip diameter value. The total count of all clips generated is shown in table I.

The NHS (Non-Hotspot) clips are generated by first randomly picking locations on the physical design which are at least a certain minimum distance away from each of the hotspot locations in the hotspot site list. This minimum distance is chosen to be 1.5 times the radius of the HSC clips. Next, square layout snippets with a given diameter are clipped from the physical design at each of these randomly picked locations. This methodology guarantees the absence of hotspots within the extent of any NHS clip. For the experiments described in this article, the total number of NHS clips selected is about 5 times the number of HSC clips. The exact number for each clip diameter is shown in table I. As shown in the table, this methodology results in a balanced dataset, with the count of HSO clips roughly equal to the count of NHS clips.

The second step in the data generation process is the conversion of the labeled layout snippets into feature vectors that can be read by a machine learning system. To generate the feature vector from a layout snippet, a previously described methodology was followed [14], [12]. Briefly, a square grid is superimposed on top of the layout clip and the polygon density is calculated within each grid cell to generate a square matrix of density values. This two dimensional square matrix is then unrolled into a one dimensional feature vector, also referred to as the density vector. Since the size of each grid cell is typically much larger than the 1 database unit (1 dbu) resolution of the layout clip, this conversion results in a down sampling of the layout clip. The bigger the size of each cell in the square grid, the higher is the extent of down sampling.

B. Model training and testing

The labeled dataset consists of density vectors and their corresponding labels - HSC, HSO or NHS. The composition of the data set for each clip diameter is as shown in table I. For all the experiments described in this article, the density vectors labeled as HSC were ignored and discarded from the data set. This was done to specifically remove any hidden bias that the HSC clips might carry. Since all the HSC clips have a line-break or pinching hotspot at the center, each clip has a metal line that runs along the centerline of the pattern.

This specific feature of all HSC clips differentiates them from clips of other classes and can be a hidden bias.

To generate the training and test data sets, the technique of cross-validation or rotation estimation was used [15]. In this technique, the dataset is split into 3 equal smaller sets. For each of the 3 sets or folds, a model is trained using the remaining 2 of the folds as training data. The resulting model is then validated on the remaining part of the data. The performance measure reported by this technique is the average of the values computed across each of the 3 folds. To report the model performance on each of the 3 folds, the scoring metric of F_1-score was used. This is a popular metric used to measure the performance of binary machine learning classifiers as it take both precision and recall into account. The F_1-score is defined as the harmonic mean of precision and recall. An F_1-score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.

TABLE II: Table with the model name and parameter value ranges used for the machine learning experiments.

Algorithm	**Parameter: [Values]**
kNN (k-Nearest Neighbor)	n_neighbors: [3-11]
SVM (Support Vector Machine)	kernel: [linear, rbf], C: [10^{-2}-10^{2}], gamma(rbf): [10^{-4}-10^{4}]
DT (Decision Tree)	max_depth: [5-19]
RF (Random Forest)	n_estimators: [8-64]

C. *Selection of model parameters*

Four popular machine learning algorithms were picked for evaluation. These include - kNN - k-Nearest Neighbors [16], SVM - Support Vector Machine [17], DT - Decision Tree, RF - Random Forest [18]. Each of these algorithms requires a set of configuration parameters to initialize the setup. For example, the kNN algorithm which works by picking the nearest neighbors in the n-dimensional feature vector space, uses a parameter to define the number of neighbors to be used for averaging. To pick the optimal set of these parameters, several models for each algorithm with parameter values spanning a large range of values were used to train and test the model on a small partition of the dataset consisting of 5000 data points. The parameters which resulted in the highest average F_1-score were picked for further experimentation.

III. Results

Several experiments were performed for each selected machine learning algorithm by varying the following parameters - the clip diameter, the extent of down sampling for density vector generation and the number of data points in the training data set. Every experiment consisted of training a machine learning model and testing it using the 3-fold cross-validation technique as described in section II-B.

A. *Visualizing data*

The process of conversion of the layout clip into a feature vector involves down sampling the clip by computing the polygonal area density within each cell of an overlaid square grid as described in section II-A. A down sampled image of the clip can be generated by assigning each grid cell an intensity value that corresponds to the density within the cell. Figure 3 shows a series of these images generated from a sample layout clip by down sampling over a square grid with varying grid size. As the size of each cell in the grid increases, the image resolution goes down and precise geometric details about the polygons in the original clip are successively lost.

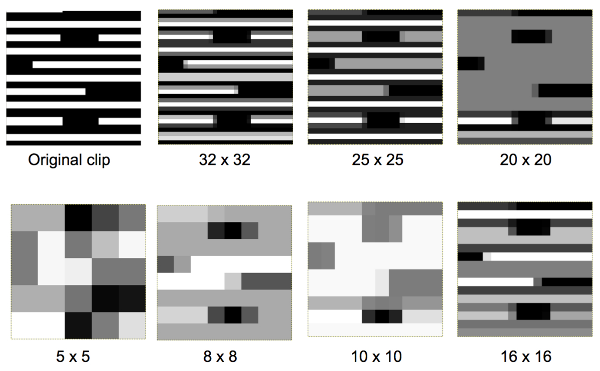

Fig. 3: Series of images showing the effect of down sampling on the original hotspot image. The image labels correspond to the number of pixels.

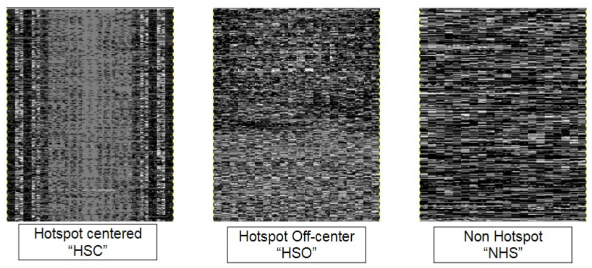

Fig. 4: Representation of the density vector for each pattern family for the large diameter clips with a feature vector length of 400. Each row of pixels corresponds to the density vector for a single pattern.

Another way to visualize the 3 clip families of HSC, HSO and NHS is to look at the image representation of the one dimensional density vector. Figure 4 shows the density vector representation of 500 layout clips arbitrarily picked from each clip family. In each of the 3 images, a row of pixels corresponds to the density vector for a single clip. Each pixel corresponds to a single feature in a single density vector with an intensity value that scales with the feature value. The image is generated by stacking rows of pixels for density vectors in the same clip family. Distinctive banding patterns can be seen in the HSC clip family, while the HSO and

NHS appear to be relatively random. The banding pattern in the HSC clip family is most likely due to the similarity in the appearance of all HSC clips which have a metal line that runs along the centerline of the clip. The random offset applied to the hotspot site appears to get rid of this banding signature by disturbing the symmetry. The NHS image is expected to appear random as it is a result of clips that have been harnessed from the physical design without any predisposition.

B. *Effect of training dataset size*

The size of the training data set is an important aspect for any machine learning model. Typically, the bigger the training data set, the more accurate a model is. Figure 5 shows the effect of increasing training data set size on the model performance for four different machine learning algorithms with the same down sampling grid cell size. For both, the small and large clip diameters, across all machine learning algorithms tested, it can be seen that the F_1-score monotonically increases as the the number of points in the training dataset increase. For the large clip diameter, the Decision Tree (DT) algorithm seems to be most sensitive to the data set size. For the small clip diameter, the sensitive across the four algorithms is roughly the same. The test results for a large dataset size are discussed in section III-D.

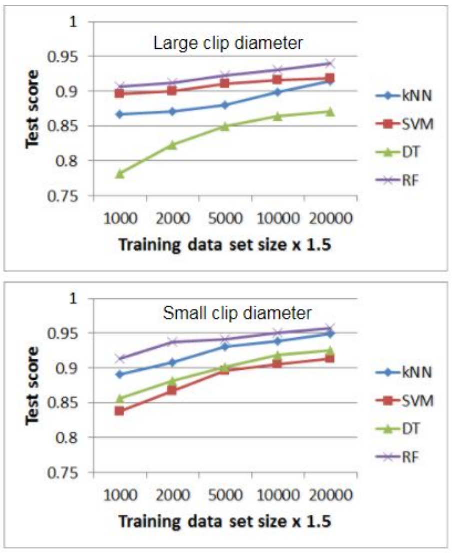

Fig. 5: Learning curves showing the variation of the test data set F_1-score as a function of total number of data points for four different machine learning algorithms at the same down sampling grid cell size.

C. *Effect of number of features*

As discussed in section III-A, the extent of down sampling directly controls the geometric information that flows from the original clip to the feature vector. The bigger the grid cell size, the higher the down sampling and the shorter the feature vector. Figure 6 shows the model performance as a function of the number of features for both the large and small clip diameter cases for a dataset with 10,000 total data points. The general trend observed across different models is that the model performance improves as additional features are added. For both clip sizes, the Support Vector Machine (SVM) algorithm is found to be most sensitive to the number of features while Random Forest (RF) is least sensitive.

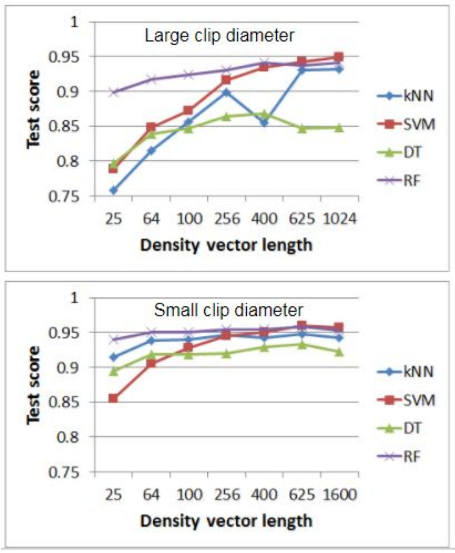

Fig. 6: Variation of test dataset F_1-score as a function of the length of the density vector for four different machine learning algorithms.

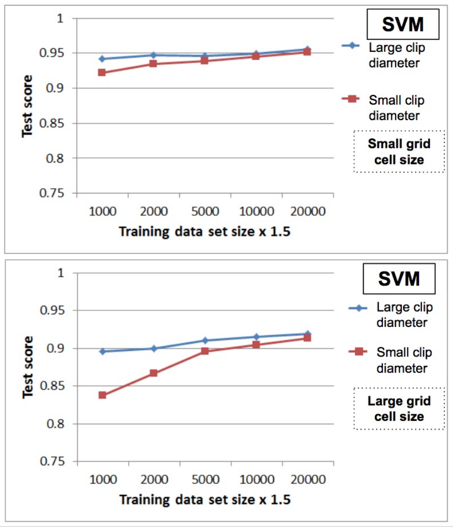

Fig. 7: Learning curves showing test dataset F_1-score as function of total number of data points for Support Vector Machine (SVM) algorithm when the clip diameter is varied.

D. *Effect of pattern size*

The clip size determines the amount of design context that is captured around the hotspot. While this is obvious for HSC class of clips, it is also true for HSO class of clips since the hotspot is contained within the central 75% of the region

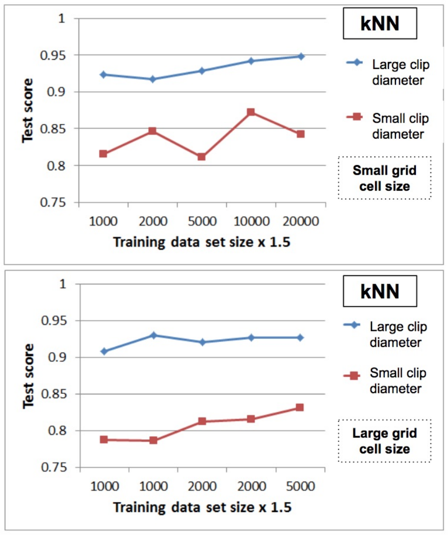

Fig. 8: Learning curves showing test dataset F_1-score as function of total number of data points for k-Nearest Neighbors (kNN) algorithm when the clip diameter is varied.

of the clip. Figures 7 and 8 show a comparison between the model performance for large and small clip diameter datasets for two machine learning algorithms - SVM (Support Vector Machine) and kNN (k-Nearest Neighbors) when the down sampling grid cell size is fixed. We find that irrespective of the algorithm and grid cell size, models trained on data from clips with a larger radius show a higher performance. While this difference is quite obvious for kNN algorithm at all dataset sizes shown in figure 8, it diminishes with increasing dataset size for SVM algorithm as shown in figure 7. Based on the results in section III-C, as the number of features per layout clip increase, the model performance improves. For a given grid cell size, a clip with a large size has more features compared to a smaller clip. Thus, a model trained on larger clip size dataset shows higher performance than the one trained on a smaller clip size dataset.

Figure 9 shows the effect of clip or pattern size on model performance for two algorithms - kNN and SVM, for the same length of feature vector. Unlike the trend in figures 7 and 8, we find that when the length of the feature vector is fixed, the models trained on clips with a smaller diameter have a higher performance than the ones trained on clips with a larger diameter. As noted in section III-C, with the increase in the number of features per layout clip, the model performance improves. The number of features or the length of the density vector can be increased by either increasing the size of the clip for a given grid cell size or by reducing the grid cell size for a given size of the clip. In the case of figure 9, to fix the number of features between different sized clips, a finer grid was used for density computation on the smaller clips. A finer grid preserves more geometric information about the pattern than a coarser grid. Despite the reduction in the design context around the hotspot in the smaller diameter clips, the additional resolution available due to the finer grid appears to boost the model performance.

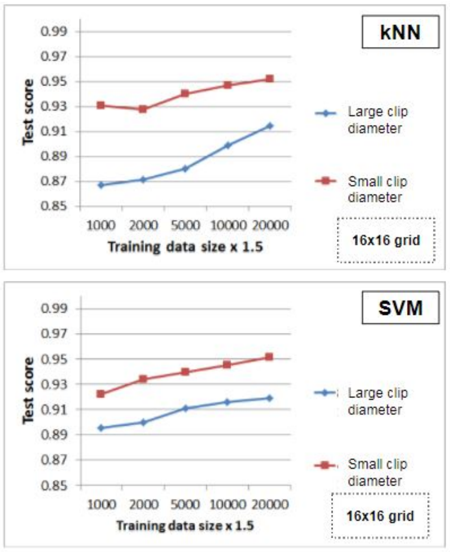

Fig. 9: Learning curves showing test data set F_1-score as function of total number of data points for Support Vector Machine (SVM) algorithm (Top) and k-Nearest Neighbors (kNN) algorithm (Bottom). Results are shown for large and small clip diameters, down sampled to 16 x 16 pixels.

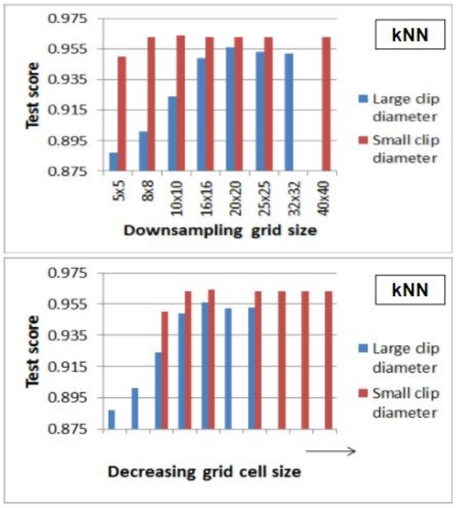

Fig. 10: Test data set F_1-score for k-Nearest Neighbor (kNN) algorithm using a balanced data set with about 500,000 data points. The plots show the comparison between large and small clip diameters.

To investigate the model performance for different clip diameters in the limit of a large dataset, we performed multiple experiments using the kNN algorithm and the full dataset of about 500,000 data points. These results are shown in figure 10. The same set of results are presented in two different ways. In figure 10, top, the model performance is plotted as a function of the count of cells in the down sampling grid. This is equivalent to the length of the feature or density vector in the given context. In figure 10, bottom, the model performance is plotted as a function of the grid cell size used for down sampling. As previously mentioned, for the same grid cell size, a clip with a larger diameter will have a longer feature vector.

We find that at each fixed grid cell size value or density vector length, the model trained on the dataset with small clip diameter has a higher performance. Upon comparison with the results with smaller datasets in figure 9, these results are consistent. For the same length of the feature vector, the smaller clips have a finer down sampling grid, yielding models with a higher performance. However, upon comparison with the results in figures 8 and 7, these results seem to be surprising. For the same grid cell size, the models trained on clips with a smaller diameter, appear to have a slightly higher performance than the models trained on clips with a larger diameter. However, the difference in F_1-score is on the order of 1-2 % and might not be statistically significant.

IV. CONCLUSIONS

We have demonstrated that machine learning can be a viable option for detecting hotspots in a physical design. By solving the critical problem of class imbalance through generation of multiple hotspot clips from a single hotspot in the design, we are able to generate a large balanced dataset and thereby build models with a very high classification accuracy. We tested four standard machine learning algorithms, viz. kNN - k-Nearest Neighbors, SVM - Support Vector Machine, DT - Decision Tree and RF - Random Forest. Once the configuration parameters of these algorithms are optimized and a sufficiently large training dataset is available, all of them work very well, with RF showing the highest performance across different dataset sizes, clip sizes and number of features. However, considering the simplicity of the kNN algorithm, it shows very good results, with an F_1-score only a few % below the RF models.

By utilizing a large balanced dataset, we have studied the effect of various parameters on the model accuracy. These parameters include - the diameter of the layout clips, the length of the density vector and the size of the training dataset. Based on our experiments, a larger training dataset size and a longer density vector, both improve the model prediction accuracy. Thus, a longer density vector with more features can compensate for a smaller training dataset size and vice versa. This complementary behavior is significant for efficient utilization of computational resources. The trends against the clip size are not as obvious in our study since we only considered two clip sizes. A more comprehensive study is needed to fully understand all the effects of clip size on prediction accuracy.

REFERENCES

[1] S. Shang, Y. Granik, and M. Niehoff, "Etch proximity correction by integrated model-based retargeting and opc flow," pp. 6730 – 6730 – 4, 2007. [Online]. Available: http://dx.doi.org/10.1117/12.746773

[2] M. Salama and A. Hamouda, "Efficient etch bias compensation techniques for accurate on-wafer patterning," pp. 9427 – 9427 – 7, 2015. [Online]. Available: http://dx.doi.org/10.1117/12.2085956

[3] P. Verma, S. Somani, Y. Y. Ping, P. Pathak, R. S. Ghaida, C. P. Babcock, F. Batarseh, J. Wang, S. Madhavan, and S. McGowan, "Hybrid opc flow with pattern search and replacement," pp. 9426 – 9426 – 8, 2015. [Online]. Available: http://dx.doi.org/10.1117/12.2087094

[4] P. Verma, F. Batarseh, S. Somani, J. Wang, S. McGowan, and S. Madhavan, "Pattern-based pre-opc operation to improve model-based opc runtime," pp. 9235 – 9235 – 11, 2014. [Online]. Available: http://dx.doi.org/10.1117/12.2068998

[5] J.-Y. Wuu, F. G. Pikus, and M. Marek-Sadowska, "Efficient approach to early detection of lithographic hotspots using machine learning systems and pattern matching," pp. 7974 – 7974 – 8, 2011. [Online]. Available: http://dx.doi.org/10.1117/12.879546

[6] J.-Y. Wuu, F. G. Pikus, A. Torres, and M. Marek-Sadowska, "Detecting context sensitive hot spots in standard cell libraries," pp. 7275 – 7275 – 9, 2009. [Online]. Available: http://dx.doi.org/10.1117/12.814316

[7] S. Mostafa, J. A. Torres, P. Rezk, and K. Madkour, "Multi-selection method for physical design verification applications," pp. 7974 – 7974 – 8, 2011. [Online]. Available: http://dx.doi.org/10.1117/12.878463

[8] D. Ding, A. J. Torres, F. G. Pikus, and D. Z. Pan, "High performance lithographic hotspot detection using hierarchically refined machine learning," in *Proceedings of the 16th Asia and South Pacific Design Automation Conference*, ser. ASPDAC '11. Piscataway, NJ, USA: IEEE Press, 2011, pp. 775–780. [Online]. Available: http://dl.acm.org/citation.cfm?id=1950815.1950963

[9] D. Ding, X. Wu, J. Ghosh, and D. Z. Pan, "lithographic hotspot detection with critical-feature extraction and classification," in *2009 IEEE International Conference on IC Design and Technology*, May 2009, pp. 219–222.

[10] D. Ding, B. Yu, J. Ghosh, and D. Z. Pan, "Epic: Efficient prediction of ic manufacturing hotspots with a unified meta-classification formulation," in *17th Asia and South Pacific Design Automation Conference*, Jan 2012, pp. 263–270.

[11] A. B. Kahng, C. H. Park, and X. Xu, "Fast dual-graph-based hotspot filtering," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 9, pp. 1635–1642, Sept 2008.

[12] Y. T. Yu, G. H. Lin, I. H. R. Jiang, and C. Chiang, "Machine-learning-based hotspot detection using topological classification and critical feature extraction," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 3, pp. 460–470, March 2015.

[13] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *In Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI*, 2000, pp. 111–117.

[14] J. Y. Wuu, F. G. Pikus, A. Torres, and M. Marek-Sadowska, "Rapid layout pattern classification," in *16th Asia and South Pacific Design Automation Conference (ASP-DAC 2011)*, Jan 2011, pp. 781–786.

[15] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection." Morgan Kaufmann, 1995, pp. 1137–1143.

[16] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879

[17] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep 1995. [Online]. Available: https://doi.org/10.1007/BF00994018

[18] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, Aug 1998.