A Stochastic Approach to Handle Non-Determinism in Deep Learning-Based Design Rule Violation Predictions

Rongjian Liang
liangrj14@tamu.edu
Department of CSE
Texas A&M University

Hua Xiang
huaxiang@us.ibm.com
IBM Research

Jinwook Jung
jinwookjung@ibm.com
IBM Research

Jiang Hu
jianghu@tamu.edu
Department of ECE & CSE
Texas A&M University

Gi-Joon Nam
gnam@us.ibm.com
IBM Research

ABSTRACT

Deep learning is a promising approach to early DRV (Design Rule Violation) prediction. However, non-deterministic parallel routing hampers model training and degrades prediction accuracy. In this work, we propose a stochastic approach, called LGC-Net, to solve this problem. In this approach, we develop new techniques of Gaussian random field layer and focal likelihood loss function to seamlessly integrate Log Gaussian Cox process with deep learning. This approach provides not only statistical regression results but also classification ones with different thresholds without retraining. Experimental results with noisy training data on industrial designs demonstrate that LGC-Net achieves significantly better accuracy of DRV density prediction than prior arts.

CCS CONCEPTS

• **Hardware → Electronic design automation**; • **Physical design (EDA)**;

KEYWORDS

design rule violation, routability, stochastic modeling, deep learning

ACM Reference Format:
Rongjian Liang, Hua Xiang, Jinwook Jung, Jiang Hu, and Gi-Joon Nam. 2022. A Stochastic Approach to Handle Non-Determinism in Deep Learning-Based Design Rule Violation Predictions. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '22), October 30-November 3, 2022, San Diego, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3508352.3549347

1 INTRODUCTION

Achieving design closure without any design rule violations (DRVs) is a fundamental requirement for VLSI designs. However, accurate design rule checking (DRC) is only possible after detailed routing, which is among the last few steps in physical design with little room to fix all the remaining DRVs. In this regard, many research efforts [2, 3, 12, 13, 16, 20, 21, 23] have been undertaken to predict DRVs at earlier design stages, e.g., placement or global routing. Deep learning-based techniques have especially become popular to deliver early and high-fidelity DRV prediction, thanks to the strong knowledge extraction and reuse capability.

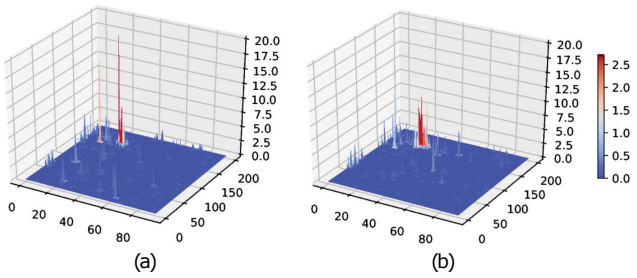

Figure 1: DRV density maps of two routing solutions for the same placement with an industrial multi-threaded detailed router.

Deep learning-based approaches typically require deterministic training data. When it comes to the DRV prediction, however, non-deterministic parallel routing can bring about non-negligible randomness in DRV distribution [15]. Figure 1 shows the DRV maps of two routing solutions on the identical placement obtained by an industrial parallel detailed router. Although they are similar to each other, the two DRV maps are far from identical. From the view of training machine-learning (ML) models, the DRV labels are very *noisy*. Even more complicated, the randomness is imbalanced across a layout; the dissimilarity of the two DRV maps in Figure 1 gets significant in regions with higher DRV density. Such noise hampers the model training and degrades the model performance. Nonetheless, to the best of our knowledge, the issue of non-deterministic training data has hardly ever been studied in the EDA field.

In this work, we propose a stochastic approach to handle the non-deterministic behavior of parallel detailed routers in deep learning-based DRV prediction. With novel DRV distribution modeling and Maximum Likelihood Estimation-based training techniques, the proposed method can predict the parameters of the probabilistic model governing the non-deterministic nature of DRV occurrence. Thus, our deep-learning model can provide more detailed DRV information in presence of non-determinism, which cannot be dealt

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ICCAD '22, October 30-November 3, 2022, San Diego, CA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9217-4/22/10…$15.00
https://doi.org/10.1145/3508352.3549347

with by most of the previous works based on binary classification. Furthermore, the prediction outcome can be transformed into DRV classification results by setting proper thresholds. It is also noteworthy that our proposed model needs single-pass training to give binary-classification results for various label thresholds, while a binary classifier requires multiple rounds of training, each with a distinct label threshold.

Our main contributions are summarized as follows:

(1) We model the stochastic spatial distribution of DRVs via a *Log Gaussian Cox (LGC) process* [5], which is a Poisson point process [6] in 2D layout space with a *Gaussian Random Field* (GRF) [14] that handles the correlation between nearby regions. We build a deep learning framework – **LGC-Net**, which is a realization of LGC process on the J-Net architecture [12].
(2) We develop *the GRF layer* – a new type of neural network layer structure. It is an implementation of the theoretic concept of GRF and designed in a way to be seamlessly integrated with our LGC-Net architecture so that spatial correlations of DRVs can be well captured.
(3) We propose a new loss function – *Focal Likelihood Loss*, for training the proposed LGC-Net model. It unifies the concepts of likelihood and focal loss, which are usually used separately. Its computation is built upon the Poisson point process in the LGC process. The likelihood estimate makes our LGC-Net training much more robust to noisy data than the popular mean-square-error-based loss functions. Moreover, the focal loss part helps deal with imbalanced training data, which is often seen in cases of DRV predictions.
(4) LGC-Net can be applied for either regression or classification. For classification, it can be trained only once and then applied with different label thresholds without retraining. This is a very appealing difference from conventional classification models.
(5) Experiments results based on noisy training data obtained from a set of industrial 7nm designs demonstrate that our method achieves stochastic DRV density prediction performance significantly better than previous arts. Our regression-based classification results are also remarkably superior to those from recent previous work.

The rest of this paper is organized as follows. Section 2 reviews relevant previous works. Section 3 presents preliminaries for understanding our method. Section 4 illustrates the details of our LGC-Net to handle non-determinism in DRV prediction. Section 5 provides experimental results, and Section 6 concludes the paper.

2 PREVIOUS METHODS

Design rule violations is a widely-adopted indicator of layout routability. DRV prediction in early design stages allows designers or tools to prevent DRVs in a proactive manner. Machine learning-based methods [2, 20], especially deep learning-based techniques [12, 21, 21, 23], have demonstrated their great potential in providing early and high-fidelity DRV prediction. A support vector machine-based technique is developed in [20] to predict locations of DRVs. A neural network-based approach is proposed in [2] for short violation prediction from a placed netlist. RouteNet [23] is the first deep learning-based DRV predictor. It utilizes a convolutional neural network (CNN) to predict the total number of DRVs and a fully convolutional network (FCN) to pinpoint the DRV locations. A CNN-based approach is developed in [21] to transform global routing reports into DRV Maps. The work of J-Net [12] proposes a customized convolutional network to pinpoint DRV locations at advanced sub-10nm process nodes, where pin accessibility becomes a major contributor to DRVs.

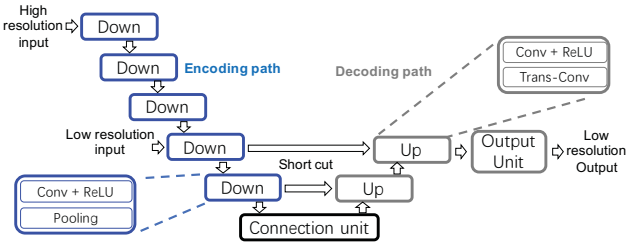

Figure 2: J-Net convolutional architecture for DRV prediction [12].

Other related works include routing congestion predictors [3, 9, 18, 22] for VLSI and FPGA circuits, a neural architecture search technique for automatic routability predictor development [7], a CNN-based classifier to identify pin accessibility risks [16] and routability predictor-driven placement optimization methods [13, 19]. However, the issue of non-deterministic training data has not yet been studied in the aforementioned works.

3 BACKGROUND

3.1 J-Net Convolutional Network

For accurate DRV prediction at advanced technology nodes, both (i) higher resolution pin configuration images and (ii) lower resolution tile-based features, e.g., net density, need to be considered. A flexible convolutional architecture, called J-Net, is proposed in [12] to handle the mixed resolution images. J-Net has an encoder-decoder architecture as shown in Figure 2. Feature maps of different resolutions are fed into different levels at the encoding path. If a feature map is fed to a middle level of the encoding path, it is concatenated with the feature representations of the same resolution produced from the previous levels. In this way, J-Net can accommodate input features with different resolutions effectively.

3.2 Focal Loss

Focal loss (FL) [17] is a loss function proposed to address extremely imbalanced data in classification model training. For a sample with label $l \in \{0, 1\}$, the raw classification outcome $y \in \mathbb{R}$ $(0 \leq y \leq 1)$ can be viewed as the probability of this sample being classified as "1." The focal loss of y is given by:

$$\mathrm{FL}(y, l) = -(1 - p_t)^{\gamma} \log(p_t), \tag{1}$$

where γ is a hyperparameter, and p_t is defined as y if $l = 1$ and $1 - y$ otherwise. The term p_t near 1 indicates samples where the model can correctly classify with high probabilities, whereas p_t near 0 means difficult-to-classify samples. When a sample has small p_t, the modulating factor $(1 - p_t)^{\gamma}$ is near 1 and the loss is unaffected. As p_t goes to 1, the modulating factor goes to 0 down-weighting the loss for well-classified samples. Hyperparameter γ adjusts the rate at which easy samples are down-weighted. Figure 3 depicts how γ affects FL. Note that the FL can not be applied to regression problems directly.

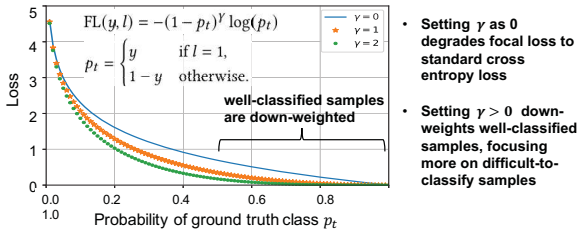

Figure 3: Illustration of focal loss [17].

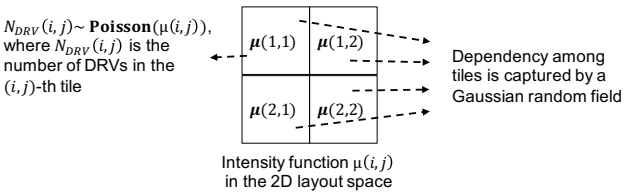

Figure 4: A Log Gaussian Cox process in a 2D layout space.

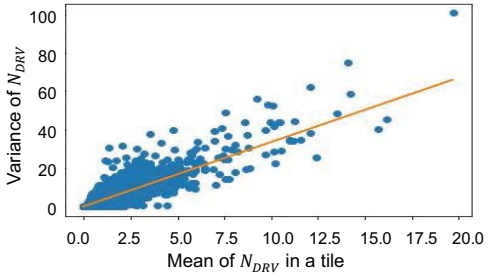

Figure 5: Mean vs. variance of the number of DRVs in a tile.

3.3 Log Gaussian Cox Process

A point process [6] is a probabilistic model for random scattering of points in a mathematical space $S \subseteq \mathbb{R}^d$ (the d-dimensional Euclidean space). A point process on S is called a Poisson point process if the number of points in any bounded region $B \subseteq S$, denoted by $N(B)$, is a random variable following a Poisson distribution. A Poisson point process is fully characterized by a location-dependent *intensity function* $\mu(B)$, which gives the expected number of points in B as well as its variance, i.e., $E[N(B)] = \mathrm{Var}[N(B)] = \mu(B)$.[1] The Poisson point process can be viewed as a fundamental point process model in that it possesses the property of "complete spatial randomness" of points.

Real-world applications of point processes usually exhibit some degree of clustering or repelling between the points in a space. The *Log Gaussian Cox (LGC) process* [5] is an extension of the Poisson point process to handle such spatial correlation between points via a hierarchical stochastic structure. An LGC process consists of a Poisson point process with its intensity function modeled by a *Gaussian Random Field* (GRF) [14]. A random field represents the joint probability distribution for a set of random variables over a multi-dimensional space such as $\mathbb{R}^2$. A GRF is a random field where every finite collection of those random variables obeys a multivariate normal distribution. In an LGC process, a GRF is leveraged to capture the pair-wise correlation between the intensity values of regions.

4 STATISTICAL DRV PREDICTION

4.1 Modeling Statistical Distribution of DRVs by Log Gaussian Cox Process

Given a circuit layout, we tessellate it into an array of uniform rectangular tiles, each of which is an $l \times l$ square;[2] a rectangular layout with size $W \times H$ is divided into $w \times h$ tiles, where $w = \lceil W/l \rceil$ and $h = \lceil H/l \rceil$. With a tessellated layout space, we model the spatial distribution of non-deterministic DRVs as an LGC process in a discrete 2D space indexed by tile row and column numbers (i, j) with intensity function $\mu(i, j)$. Figure 4 illustrates such stochastic modeling of non-deterministic DRVs. The Poisson point process implies that the number of DRVs occurring at tile (i, j), denoted by $N_{\mathrm{DRV}}(i, j)$, obeys a Poisson distribution with mean $\mu(i, j)$, i.e.,

$$\Pr\left[N_{\mathrm{DRV}}(i, j) = k\right] = e^{-\mu(i,j)} \frac{\mu(i, j)^k}{k!}, \quad k = 0, 1, 2, \ldots \qquad (2)$$

[1] Formally, $\mu(B)$ is defined with a locally integrable function $\lambda : \mathbb{R}^d \to [0, \infty)$ such that $\mu(B) = \int_B \lambda(x)dx$. If this integral is finite, $N(B)$ obeys a Poisson distribution, i.e., $\Pr[N(B) = k] = e^{-\mu(B)} \frac{\mu(B)^k}{k!}$, where k is a non-negative integer.

[2] In our experiments, l is set to be $1.26\mu m$.

and GRF captures dependency among tiles by enforcing $\mu(i, j)$ to have correlation with nearby layout tiles.

We argue that the LGC process can be a good match to the non-deterministic DRV distribution. The domain of an LGC process is the set of non-negative integers, which matches the range of $N_{\mathrm{DRV}}(i, j)$ in the layout naturally. To further demonstrate the rationale behind adopting an LGC process for modeling non-deterministic DRV distribution, we took five placed circuits and obtained DRV maps from 32 detailed routing runs for each circuit with an industrial router. Treating $N_{\mathrm{DRV}}(i, j)$ as a random variable, we obtained 32 samples for each random variable from which we estimate their mean and variance. Figure 5 shows the relationship between the mean and variance of $N_{\mathrm{DRV}}(i, j)$. According to the LGC process, the mean is supposed to be equal to the variance. We can find that the variance has a roughly linear relationship with the mean, which conforms to the LGC process characteristics. Although we observe the variance-to-mean ratio being about 3, a relatively-small ratio shows that the LGC process is a reasonable model for the DRV distribution. In addition, we can see clustering behaviors of DRV occurrence in Figure 1, which can be easily captured in our stochastic modeling thanks to GRF in the LGC process.

4.2 Problem Formulation of LGC-Net

Given a pre-route design, our goal is to predict the intensity function $\mu(i, j)$ of each layout tile (i, j), which we name the *DRV intensity*. According to the LGC process, $\mu(i, j)$ is equal to the mean and variance of the DRV density on tile (i, j). Hence, with predicted $\mu(i, j)$ for each tile (i, j), we can derive the Poisson distributions governing non-deterministic DRVs.

We note that DRVs are caused mainly by pin accessibility issues [4] and routing congestion. Hence, we use the following feature images as input features of the proposed model.

(1) **High resolution pin configuration images**. These images capture the exact locations and shapes of every pin in the

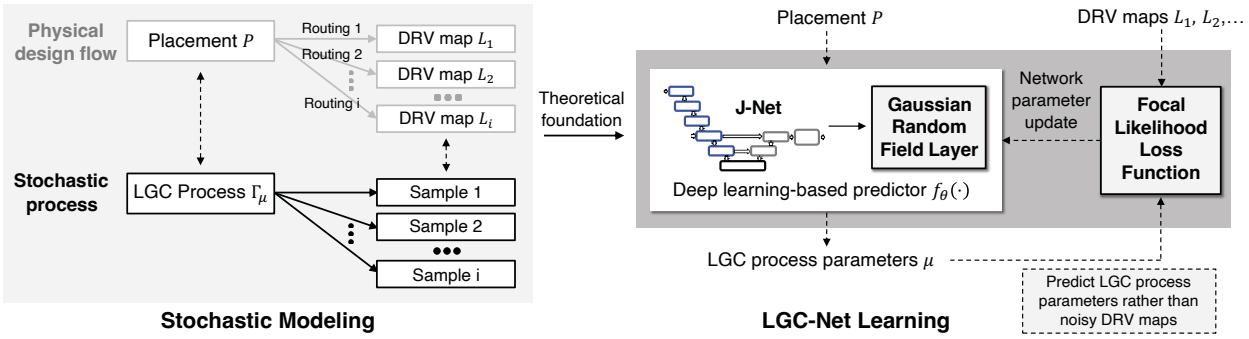

Figure 6: Overview of the proposed LGC-Net to handle non-deterministic routing solutions in DRV prediction.

entire layout [12]. For each metal layer where pins reside, one pin configuration image is generated. These pin configuration images help identify pin accessibility issues. In our setup, a tile (1.26×1.26 μm^2) corresponds to 126×126 pixels in a pin configuration image.

(2) **Tile-based layer-wise routing congestion maps**. They are produced by a trial global router. Unlike a majority of prior arts resorting to a 2D routing congestion map, we use 3D layer-wise congestion maps as input, since the actual routing is conducted in a 3D space. Although a few recent efforts [7, 12] *classify* DRC hotspot without any routing information, predicting DRV intensity with noisy label data is evidently much more challenging. Besides, it is reasonably fast to call a trial global router, which takes a couple of minutes for one placement instance on our test cases.

We now state the target problem as follows. Given a placement with its trial routing result, predict a 2D DRV intensity map in which each entry (i, j) represents the DRV intensity $\mu(i, j)$ in the corresponding layout tile. Labels are extracted from post-routing design rule checking.

4.3 Overview of LGC-Net

Figure 6 depicts an overview of the proposed *LGC-Net*. We view that each placed layout corresponds to one LGC process describing the statistical distribution of DRVs. The similar but not identical DRV labels from different routing solutions for the same placement are regarded as samples of the corresponding LGC process. We integrate a deep neural network J-Net [12] with our proposed GRF layer into LGC-Net for mapping a placement instance to the parameters (i.e., the DRV intensity function) of the corresponding LGC process. The GRF layer is a customized network layer that implements a Gaussian random field, allowing our LGC-Net to capture the spatial correlations of DRVs. Based on the LGC process, we develop the Focal Likelihood Loss (FLL) function, which guides the training of LGC-Net in a Maximum Likelihood Estimation (MLE) manner. This is another key difference that distinguishes LGC-Net from J-Net and other deep learning models.

4.4 Gaussian Random Field (GRF) Layer

The GRF layer is a customized neural network layer implementation of GRF that can be integrated with deep learning models to achieve end-to-end training. It processes the *unstructured* prediction output $\mathbf{Z} \in \mathbb{R}^{w \times h}$ of the J-Net model and outputs a *structured* output $\mathbf{Y} \in \mathbb{R}^{w \times h}$. Here, unstructured output means the intensity function prediction outcome without considering the dependency among tiles, while the structured output means the prediction taking correlation among nearby tiles into account.

The GRF layer is essentially composed of a set of matrix operations, whose computation can be conveniently accelerated by off-the-shelf ML packages. Denote the flattened vector of $\mathbf{Z}$ as $\mathbf{z}$, which is a vector of length $w \times h$, and the flattened vector of $\mathbf{Y}$ as $\mathbf{y}$. The GRF layer outputs $\mathbf{z}$ that maximizes the probability

$$\Pr[\mathbf{y} \mid \mathbf{z}] = \frac{1}{\alpha} e^{-E(\mathbf{y},\mathbf{z})}, \tag{3}$$

where α is a normalization term ensuring that the integral of the probability over $\mathbf{y}$ is equal to 1, and $E(\mathbf{y}, \mathbf{z})$ is defined as:

$$E(\mathbf{y}, \mathbf{z}) = \sum_i (z_i - y_i)^2 + \beta \sum_{i<j} S_{i,j} (y_i - y_j)^2 . \tag{4}$$

In Equation (4), β is a weighting factor, and $S_{i,j}$ is a similarity measure between the i-th and the j-th entry. The first term on the right-hand side of Equation (4) enforces the structured output $\mathbf{Y}$ to be close to the unstructured output $\mathbf{Z}$, while the second term enforces similar entries to have similar outputs. The similarity measure $S_{i,j}$ describes the similarity among entries, which is given by $e^{-d(i,j)/\sigma}$ if $i \neq j$ and 0 otherwise, where $d(i, j)$ is the Manhattan distance between two layout tiles corresponding to entry i and j, and σ is a learnable parameter.

$E(\mathbf{y}, \mathbf{z})$ can be rewritten in a vector form as follows:

$$E(\mathbf{y}, \mathbf{z}) = \mathbf{y}^T \mathbf{Q} \mathbf{y} - 2\mathbf{y}^T \mathbf{z} + \mathbf{z}^T \mathbf{z} \propto \mathbf{y}^T \mathbf{Q} \mathbf{y} - 2\mathbf{y}^T \mathbf{z}, \tag{5}$$

where $\mathbf{Q}(i, j)$ is given by

$$\mathbf{Q}(i, j) = \begin{cases} 1 + \beta \sum_{h \neq i} S_{i,h} & \text{if } i = j, \\ -S_{i,j} & \text{if } i \neq j. \end{cases} \tag{6}$$

We let the weighting factor β be a learnable parameter.

Once the values of β and $S_{i,j}$ are obtained, $\mathbf{Q}$ can be computed. For positive-definite matrix $\mathbf{Q}$, the optimal structured output $\mathbf{y}^*$ that maximizes $\Pr[\mathbf{y} \mid \mathbf{z}]$ can be obtained by

$$\mathbf{y}^* = \mathbf{Q}^{-1} \mathbf{z} . \tag{7}$$

With the GRF layer, we use a two-stage training process. At the first stage, the GRF layer does not come into play and the J-Net part is trained solely to output unstructured prediction. At the second

stage, the GRF layer is attached to the model as the last layer and the entire model is trained end-to-end to deliver structured output. We empirically find that such two-stage training process is easier to converge than a one-stage end-to-end training.

4.5 Focal Likelihood Loss (FLL)

FLL is a novel loss function that combines the capability of focal loss to address imbalanced data and the strength of MLE (Maximum Likelihood Estimation) in coping with uncertainty. Its computation is built upon the Poisson point process in LGC process. MLE estimates the parameters of an assumed probability distribution given some observed data, by maximizing a likelihood function so that the observed data is most probable. Here, the probability distribution is the LGC process and the parameters are the DRV intensities μ; the observed data is the noisy DRV labels $\mathbf{L} \in \mathbb{N}^{w \times h}$ ($\mathbb{N}$ is the set of non-negative integers) collected after detailed routing. The likelihood is calculated as

$$\zeta(L_{i,j}, Y_{i,j}) = \log\left(\Pr[L_{i,j}|\mu(i,j) = Y_{i,j}]\right), \tag{8}$$

where the conditional probability $\Pr[L_{i,j}|\mu(i,j) = Y_{i,j}]$ is calculated according the properties of Poisson point process:

$$\Pr[L_{i,j}|\mu(i,j) = Y_{i,j}] = \frac{Y_{i,j}^{L_{i,j}} e^{-Y_{i,j}}}{L_{i,j}!}. \tag{9}$$

It is obtained by letting $\mu(i,j) = Y_{i,j}$ and $k = L_{i,j}$ in Equation (2).

FLL is defined as follows:

$$\mathrm{FLL}(\mathbf{Y}, \mathbf{L}) = -\frac{1}{wh} \sum_{i,j} \left(1 - \Pr[L_{i,j}|\mu(i,j) = Y_{i,j}]\right)^{\gamma} \zeta(L_{i,j}, Y_{i,j}). \tag{10}$$

Essentially, Equation (10) is obtained via replacing the p_t in Equation (1) by $\Pr[L_{i,j} \mid \mu(i,j) = Y_{i,j}]$.[3] FLL guides the training process to converge to a solution that maximizes the likelihood $\zeta(L_{i,j}, Y_{i,j})$. The term $1 - \Pr[L_{i,j} \mid \mu(i,j) = Y_{i,j}]$ is the focal loss term, which down-weights easy samples adaptively during training, as introduced in Section 3. The introduction of the focal loss term helps FLL handle the issue of imbalanced data sets.

In many applications including DRV intensity regression, maximizing likelihood matters more than minimizing absolute errors. Consider two cases, where one has prediction $Y_{i,j} = 0$ and label $L_{i,j} = 1$, and the other has prediction $Y_{k,l} = 9$ and label $L_{k,l} = 10$. Although both cases have the same absolute errors (or MSEs), the first case is arguably worse than the second one. Correspondingly, our FLL penalizes the first case much more heavily, since $L_{i,j} = 1$ can never happen if the Poisson distribution has mean 0. As shown in Figure 1, the absolute difference between two different routing solutions for the same design is larger in regions with large number of DRVs. As such, we prefer to tolerate larger absolute prediction errors for tiles with higher DRV density. Our proposed FLL can handle the DRV density-dependent noise nicely.

To reduce runtime overhead during training, we pre-calculate $\log(k!)$ for every integer $k \in [0, K]$ before training and store them in a look-up table. Then, Equation (8) can be calculated as:

$$\zeta(L_{i,j}, Y_{i,j}) = L_{i,j} \log(Y_{i,j}) - Y_{i,j} - \log(L_{i,j}!), \tag{11}$$

[3] In our experiments, we set γ as 1.

Table 1: Testcase characteristics.

	Design	Dimensions (μm)	#IPs	#Gates	#Nets	#DRVs
Train set	D1	161.28 × 72.58	2	26906	26647	330142
	D2	122.88 × 124.42	0	44088	46461	33133
	D3	142.08 × 124.42	0	76799	79993	17795
	D4	103.68 × 305.86	0	126238	131862	268615
	D5	138.24 × 305.86	0	149351	151128	6394
	D6	276.48 × 510.62	16	172133	159509	15471
	D7	249.60 × 300.67	3	183544	192820	84620
	D8	760.32 × 189.22	600	270614	290613	370
Test set	D9	207.36 × 145.15	0	79062	85635	192862
	D10	215.04 × 139.97	0	84477	92054	3280
	D11	211.20 × 419.90	4	217821	218583	99322

where $\log(L_{i,j}!)$ is obtained via the look-up table, and $\Pr[L_{i,j} \mid Y_{i,j}]$ is obtained by performing an exponentiation operation.

5 EXPERIMENTAL VALIDATION

5.1 Testcases and Data Collection

Experiments were conducted on 11 industrial designs at a 7nm process node, whose characteristics are summarized in Table 1. The testing designs were totally unseen during training. A total of 10 placement instances were generated for each design by varying tool parameters in an industrial synthesis & physical design flow. For each placement instance, 10 detailed routing runs were performed with the same setting. We observed around ±10% difference in the number of DRVs across different routing runs for the same placement solution. The average number of DRVs for each design are shown in Table 1. We extracted features from the design flow via *tcl* scripts and implemented our LGC-Net on the ML framework *PyTorch*. During training, a pair of (i) placement and (ii) DRV labels from one routing run was regarded as one data point. Also, we deployed two data augmentation techniques proposed in [12], i.e., random flipping and random cropping, to enlarge training set. In evaluation, we compared the prediction outcome with the average DRV ground truth across 10 routing runs.

5.2 Performance Metrics

5.2.1 Regression Metrics. We used the following metrics to assess regression performance:

(1) The mean-squared-error (MSE).
(2) The Pearson correlation coefficient between the DRV density ground truth data and the predicted DRV intensity.
(3) The mean of the predicted DRV intensity for tiles with label $N_{\mathrm{DRV}} = 0$. For perfect prediction, the mean is 0. The smaller the mean, the better the prediction performance.
(4) The mean of normalized errors for tiles with DRVs. For a tile with label l and prediction output y, the normalized error is given by $|l - y|/\sqrt{l}$. According to the Poisson processes, $\sqrt{l}$ can be viewed as an approximate of the standard deviation of the DRV density. The normalized error measures the prediction error normalized against the standard deviation.

5.2.2 Multi-Class Classification Metrics. For K-class classification, we firstly calculated the confusion matrix, where $c_{i,j}$ represents the number of tiles that are labeled class i and predicted class j. Then the following metrics were calculated:

- True-positive-rate (TPR) for class i: $\mathrm{TPR}(i) = c_{i,i} / \sum_{j=0}^{K-1} c_{i,j}$.

Table 2: Average DRV density regression performances of LGC-Net with different input features and of [11] on the test set.

Performance metrics	LGC-Net w/ different inputs				Zhou[11]
	F1	F2	F3	F4	
Correlation coefficient	0.25	0.29	0.44	0.70	0.23
MSE	3.13	0.56	0.57	0.32	4.16
Mean outcomes (DRV-free tiles)	1.76	0.27	0.33	0.06	2.03
Mean norm. error (tiles w/ DRVs)	2.78	0.93	1.30	0.86	3.31

- Precision for class i: $\text{Precision}(i) = c_{i,i} / \sum_{j=0}^{K-1} c_{j,i}$.
- F1-score for class i: $\text{F1}(i) = 2 \times \frac{\text{TPR}(i) \times \text{Precision}(i)}{\text{TPR}(i) + \text{Precision}(i)}$.
- Macro-averaged F1-score: $\text{MacroF1} = \sum_{i=0}^{K-1} \text{F1}(i)/K$.
- Micro-averaged F1-score: $\text{MicroF1} = \sum_{k=0}^{K-1} c_{k,k} / \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} c_{i,j}$.

5.2.3 Binary Classification Metrics. For the binary classification performance, we plotted ROC (Receiver Operating Characteristic) curves dictating the trade-off between TPR and false-positive-rate (FPR). TPR is the number of correctly classified positive samples out of the total number of positive samples, and FPR is the number of negative samples wrongly predicted as positive out of the total number of negative samples.

5.3 Experiment 1: Impact of Non-Determinism

We study how the DRV prediction is affected by non-deterministic routing. For the placement instances from testing designs, the DRV label from the first routing run is compared with the average DRV label across the remaining 9 runs. The MSE is 0.24 and the correlation coefficient is 0.78. The significant error and limited correlation imply that the predictability of DRV density is greatly affected by the non-determinism in routing. It is impossible to get perfect estimation even with the time-consuming detailed routing.

5.4 Experiment 2: Regression Results and Ablation Study

We present the regression results of our LGC-Net with the following four different input feature combinations:

(1) **F1:** only 3D layer-wise congestion maps.
(2) **F2:** pin configuration maps and a few tile-based connection features extracted without trial routing results (same as [12]).
(3) **F3:** pin configuration and 2D congestion maps.
(4) **F4:** pin configuration and 3D layer-wise congestion maps.

The results from different input feature sets on DRV density regression performance are listed in Table 2. We can see that both the pin configuration images (F4 results vs. F1 ones) and the congestion maps (F3 results vs. F2 ones) are of great importance for DRV prediction. Also, comparing F4 results with F3 ones we can find that the 3D layer-wise congestion maps provide more information than the 2D congestion map.

Figure 7 shows a snapshot of our DRV prediction results with feature set F4. We can see that our prediction outcome matches well with the average DRV density map and the variance map. Our prediction outcome seems to be a smoothed version of the average DRV density map.

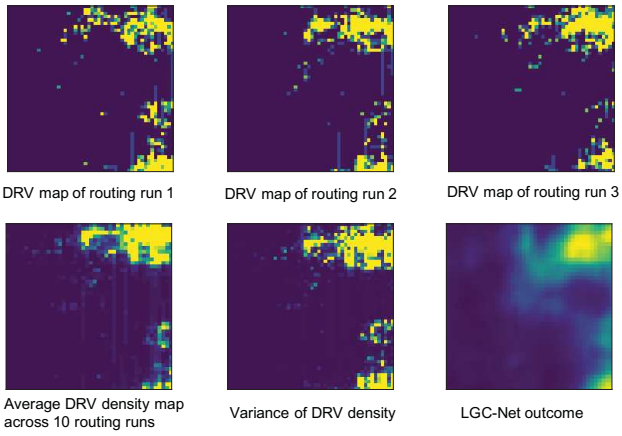

Figure 7: Illustration of regression results of D9, where the LGC-Net outcome (lower right) matches with both the mean (lower left) and variance of #DRVs (lower middle).

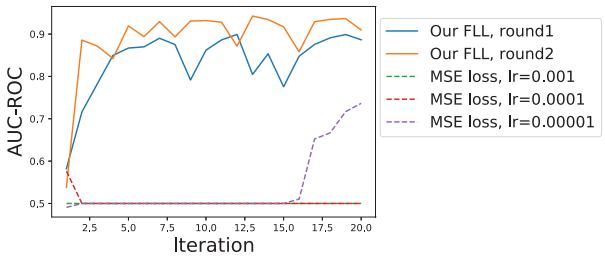

Figure 8: Validation performances for LGC-Net regression using our focal likelihood loss compared with using various MSE losses.

During training, we hold 20% training data as the validation set and evaluate the models after each training iteration. We transform the regression outcome into binary classification during validation and use the area under ROC curve (AUC-ROC) as the performance metric (the larger the area, the better the performance). To study the effectiveness of our proposed focal likelihood loss, we compare the validation performance for LGC-Net regression using our FLL with the performances using various MSE losses. As shown in Figure 8, we observe that none of the MSE loss training can converge well and they often get stuck at a solution with 0.5 area, which means random guess. The failure of MSE loss training is mainly caused by the noisy and imbalanced data set of DRV density maps. In contrast, our FLL helps the model successfully converge to meaningful solutions. In other words, our FLL technique facilitates the training of deep learning models with noisy and imbalanced data set.

We also investigate the impacts of our Gaussian Random Field layer on DRV regression. We observe that the correlation coefficient drops by 0.05 and the MSE increases by 0.07 when removing the GRF layer from our LGC-Net. The training time is doubled when integrating the LGC layer to LGC-Net due to our two-stage training scheme introduced in Section 4.4 while the inference time overhead is neglectable.

5.5 Experiment 3: Comparison with [11]

We compare LGC-Net with [11], which is the only previous DRV regression work to the best of our knowledge. It is a multivariate adaptive regression-based method using pin density, ratio of blocked areas, HPWL density and global routing results as inputs. Since [11] does not reveal its loss function, we adopt the MSE loss function in this experiment.

As shown in Table 2, LGC-Net with feature set F4 significantly outperforms the result of Zhou [11]. The superior performance from our method is brought by our input features, the deep neural work architecture, and the proposed stochastic techniques for handling noisy training data.

5.6 Experiment 4: Comparison with J-Net [12] and RouteNet [23]

In this experiment, we compare the performance of our LGC-Net with J-Net [12] and RouteNet [23], which are two state-of-the-art deep learning-based DRV classifiers. The works J-Net [12] and RouteNet [23] were originally proposed for binary DRC hotspot prediction problem. Here, we also extend them for multi-class classification, using the weighted MSE loss during training. We transform our LGC-Net regression results into classification ones by setting different thresholds. For multi-class classification, tiles are divided into three classes according to their number of DRVs, using 0.5 and 4.5 as the thresholds. Note that we use the average DRV map across 10 routing runs as ground truth in evaluation, so the thresholds are real numbers. Tiles with #DRV<0.5 are easy to route while those with > 4.5 DRVs can be viewed as hard to route. For binary classification, we compare the prediction accuracy at label thresholds 0.5, 2.5 and 4.5, separately.

5.6.1 Multi-Class Classification Results. Table 3 compares the multi-class classification performances of our LGC-Net and extensions of J-Net [12] and RouteNet [23]. It can be seen that our LGC-Net performs obviously better than other methods, by 17% higher micro-averaged F1 score and 25% higher macro-averaged F1 score. Besides, for those misclassified tile samples by our method, most of them are classified to be categories similar to their label categories. In contrast, many misclassified samples by the extended version of [12] are classified to be categories far away from their labels. As a matter of fact, it classifies over 169k class-0 samples into class 2.

The superior performance of the LGC-Net regression-based classification is attributed to two main reasons. The first lever is our stochastic techniques in handling noisy data. Also, regression labels are naturally more informative than classification labels. The LGC-Net regression-based classification can take advantage of the regression labels.

5.6.2 Classifications with Different Thresholds. ROC curves of LGC-Net regression-based binary classification, J-Net [12] and RouteNet [23] are shown in Figure 9. Compared with the J-Net and RouteNet binary classifiers, our regression-based classification delivers superior performance at various label thresholds. Another advantage of our method is that the regression model needs one-time training to give binary-classification results for various label thresholds; whereas one binary classifier is required to be trained for each distinct label threshold, consequently leading to several times longer training and inference overhead.

Table 3: Average multi-class classification results on the test set.

	Our LGC-Net		
Predicted class	0	1	2
Label=0	**932275**	80715	4367
Label=1	16170	**23765**	6254
Label=2	1239	6717	**12658**
Micro-averaged F1	**89.4%**		
Macro-averaged F1	**60.9%**		
	Extension of JNet [12]		
Predicted class	0	1	2
Label=0	**766738**	80829	169790
Label=1	10154	**3487**	32548
Label=2	1199	128	**19287**
Micro-averaged F1	**72.8%**		
Macro-averaged F1	**35.6%**		
	Extension of RouteNet [23]		
Predicted class	0	1	2
Label= 0	**633462**	73914	309981
Label= 1	15702	**8026**	20850
Label= 2	7009	5867	**9349**
Micro-averaged F1	**60.0%**		
Macro-averaged F1	**31.0%**		

5.6.3 Computing Runtime. The training time of LGC-Net on a Nvidia GeForce RTX 2080-Ti is about 60 hours, which is 2× longer than that of a J-Net classifier [12]. The longer traing time is caused by the two-stage training process introduced in Section 4.4. Since our trained model can be reused for different placements, the amortized training cost is limited. LGC-Net can be applied for either regression or classification. For classification, it can be trained only once and then applied with different label thresholds without retraining. The LGC-Net inference runtime on one placement instance is about 1 minute.

5.7 Experiment 5: Effect of Data Pruning

We study the impact of using the DRV label from single routing run or the average label across 10 routing runs for each placement during training. Arguably, the results from single routing round or the average results are deterministic, since each placement instance corresponds to one DRV label. We also try performing Gaussian filtering with a standard deviation of 1 on the noisy DRV label from one routing solution and then use it for training. We compare their prediction performances with those of using the results from all 10 routing runs.

Figure 10 shows that it does not help eliminate the negative effects of non-deterministic parallel routing by using only the result from one detailed routing round (10% data) or the average result across 10 rounds (average of 10 runs) for each placement instance. In contrast, it obviously degrades the prediction performance since less training data is deployed. In addition, simply performing Gaussian filtering on noisy DRV labels (10% data + GF) is not sufficient to handle the noisy data issue. It is also interesting to note that, Gaussian filtering enhances the performance of [12] but does not help

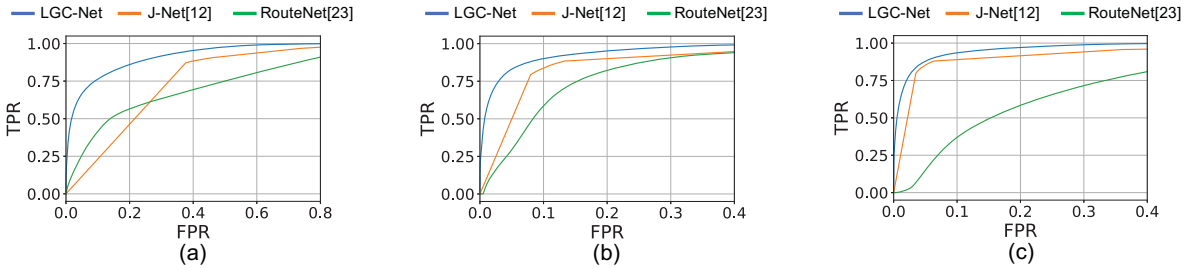

Figure 9: ROC curves for binary-classification at different label thresholds: (a) th = 0.5, (b) th = 2.5, and (c) th = 4.5.

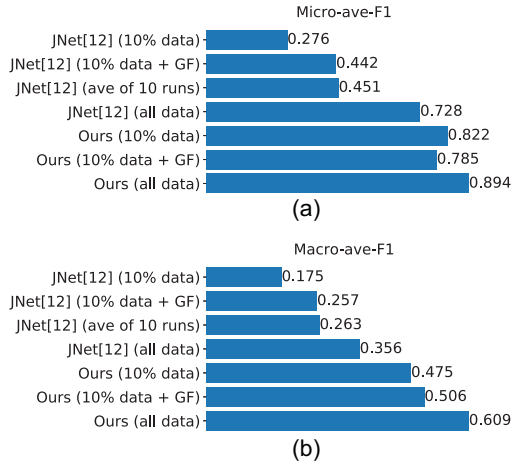

Figure 10: The effects of training (a) with one detailed routing result and (b) with the average result across 10 runs for each placement.

our approach which has already explicitly considered the stochastic properties of DRVs.

5.8 Discussion of Data Efficiency

We want to highlight that our method has better data efficiency for noisy data sets compared to previous deep learning-based DRV prediction methods. In particular, compared with [12], which utilizes a model with similar amount of learnable parameters to ours, our method has obviously better performance after training with the noisy DRV data, as shown in Figure 10.

It is noteworthy that our LGC-Net also works well when it is trained with one noisy DRV label for each placement instance (ours with 10% data in Figure 10). With the noisy data set, LGC-Net trained with one routing run result for each placement outperforms [12] trained with 10 routing run results for each placement.

6 CONCLUSION AND BROAD IMPACT

In this work, we present a stochastic approach – LGC-Net, to handle non-determinism in deep learning-based DRV predictions. Experiments results from noisy training data on industrial designs demonstrate that our method can handle the non-deterministic training data caused by parallel routing and achieves significantly better DRV density prediction performance than previous arts.

We view that the noisy label data due to non-deterministic parallel computing is a general problem in applying ML techniques to EDA problems, given that many EDA algorithms are accelerated by multi-threading. More broadly, it has been recognized that *noise* or *chaotic behavior* is ubiquitous in EDA tools [1, 8, 10, 15], i.e., a slight change in inputs or even the exact same input can lead to large variation in results. Thus, the results here provide a helpful experience for addressing this general challenge in future research.

ACKNOWLEDGMENTS

This work is partially supported by SRC GRC-CADT 3103.001 and NSF CCF-2106725.

REFERENCES

[1] A. B. Kahng and S. Mantik. 2002. Measurement of inherent noise in EDA tools. In *Proc. ISQED*. 206–211.
[2] A. F. Tabrizi *et al.* 2018. A machine learning framework to identify detailed routing short violations from a placed netlist. In *Proc. DAC*. 1–6.
[3] J. Chen, J. Kuang, G. Zhao, D. Huang and E. Young. 2020. PROS: A plug-in for routability optimization applied in the state-of-the-art commercial eda tool using deep learning. In *Proc. ICCAD*. 1–8.
[4] J. Lou, S. Thakur, S. Krishnamoorthy, and H.-S. Sheng. 2002. Estimating routing congestion using probabilistic analysis. *IEEE TCAD* 21, 1 (2002), 32–41.
[5] J. Møller, A. R. Syversveen and R. P. Waagepetersen. 1998. Log gaussian cox processes. *Scand. Stat. Theory Appl.* 25, 3 (1998), 451–482.
[6] J. Møller and R. P. Waagepetersen. 2003. *Statistical Inference and Simulation for Spatial Point Processes.* CRC Press.
[7] J. Pan *et al.* 2020. Automatic Routability Predictor Development Using Neural Architecture Search. *arXiv preprint arXiv:2012.01737* (2020).
[8] K. Jeong and A. B. Kahng. 2010. Methodology from chaos in IC implementation. In *Proc. ISQED*. 885–892.
[9] M. B. Alawieh, W. Li, Y. Lin, L. Singhal, M. A. Iyer, and D. Z. Pan. 2020. High-definition routing congestion prediction for large-scale FPGAs. In *Proc. ASP-DAC*. 26–31.
[10] M. R. Hartoog. 1986. Analysis of placement procedures for VLSI standard cell layout. In *Proc. DAC*. 314–319.
[11] Q. Zhou, X. Wang, Z. Qi, Z. Chen, Q. Zhou and Y. Cai. 2015. An accurate detailed routing routability prediction model in placement. In *Proc. ASQED*. 119–122.
[12] R. Liang *et al.* 2020. DRC hotspot prediction at sub-10nm process nodes using customized convolutional network. In *Proc. ISPD*. 135–142.
[13] S. Liu, Q. Sun, P. Liao, Y. Lin and B. Yu. 2021. Global placement with deep learning-enabled explicit routability optimization. *Proc. DATE*, 1821–1824.
[14] S. Z. Li. 2009. *Markov Random Field Modeling in Image Analysis.* Springer.
[15] Synopsys. 2021. IC Compiler II Implementation User Guide S-2021.06-SP3.
[16] T. C. Yu *et al.* 2020. Pin Accessibility Prediction and Optimization With Deep-Learning-Based Pin Pattern Recognition. *IEEE TCAD* 40, 11 (2020), 2345–2356.
[17] T. Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár. 2017. Focal loss for dense object detection. In *Proc. ICCV*. 2980–2988.
[18] R. Kirby *et al.* 2019. CongestionNet: Routing congestion prediction using deep graph neural networks. In *Proc. VLSI-SoC*. IEEE, 217–222.
[19] Y. H. Huang *et al.* 2019. Routability-driven macro placement with embedded CNN-based prediction model. In *Proc. DATE*. IEEE, 180–185.
[20] W. T. Chan, P. H. Ho, A. B. Kahng, and P. Saxena. 2017. Routability optimization for industrial designs at sub-14nm process nodes using machine learning. In *Proc. ISPD*. 15–21.
[21] W. T. Hung *et al.* 2020. Transforming global routing report into DRC violation map with convolutional neural network. In *Proc. ISPD*. 57–64.
[22] C. Yu and Z. Zhang. 2019. Painting on placement: Forecasting routing congestion using conditional generative adversarial nets. In *Proc. DAC*. 1–6.
[23] Z. Xie *et al.* 2018. RouteNet: routability prediction for mixed-size designs using convolutional neural network. In *Proc. ICCAD*. 1–8.