Encoder-Decoder Networks for Analyzing Thermal and Power Delivery Networks

VIDYA A. CHHABRIA, University of Minnesota, USA
VIPUL AHUJA, Qualcomm Technologies Inc., India
ASHWATH PRABHU, Qualcomm Technologies Inc., India
NIKHIL PATIL, Qualcomm Technologies Inc., India
PALKESH JAIN, Qualcomm Technologies Inc., India
SACHIN S. SAPATNEKAR, University of Minnesota, USA

Power delivery network (PDN) analysis and thermal analysis are computationally expensive tasks that are essential for successful IC design. Algorithmically, both these analyses have similar computational structure and complexity as they involve the solution to a partial differential equation of the same form. This paper converts these analyses into image-to-image and sequence-to-sequence translation tasks, which allows leveraging a class of machine learning models with an encoder-decoder-based generative (EDGe) architecture to address the time-intensive nature of these tasks. For PDN analysis, we propose two networks: (i) IREDGe: a full-chip static and dynamic IR drop predictor and (ii) EMEDGe: electromigration (EM) hotspot classifier based on input power, power grid distribution, and power pad distribution patterns. For thermal analysis, we propose ThermEDGe, a full-chip static and dynamic temperature estimator based on input power distribution patterns for thermal analysis. These networks are transferable across designs synthesized within the same technology and packing solution. The networks predict on-chip IR drop, EM hotspot locations, and temperature in milliseconds with negligibly small errors against commercial tools requiring several hours.

1 INTRODUCTION

1.1 Problem statement

As a consequence of aggressive technology scaling, on-chip power density has been on an increasing trend. This has led to major challenges in designing integrated circuits (ICs) in advanced technology nodes. For successful IC design, it is extremely critical to build tools that can analyze the impact of high power densities. Fast and accurate analysis that aids quick design turnaround is particularly significant for two critical and time-consuming simulations that are performed several times during the design cycle:

- *Thermal analysis*, which checks the feasibility of a placement/floorplan solution by computing on-chip temperature distributions to check for temperature hot spots.
- *Power distribution network (PDN) analysis*, which diagnoses the goodness and reliability of the PDN by checking (i) whether the voltage (IR) drops from the power pads to the gates are within specified limits, and (ii) whether the wire current densities satisfy reliability constraints related to electromigration (EM).

However, one of the major challenges with these analyses is the overhead of extremely large runtimes. The underlying computational engines that form the crux of both analyses are similar: both simulate networks of conductances and current/voltage sources by solving a large system of equations of the form $G\mathbf{V} = \mathbf{J}$ [41, 44] with millions to billions of variables. In modern industry designs, a single full-chip temperature or IR drop simulation, can take several hours. Accelerating these analyses opens the door to optimizations that iteratively invoke these engines under the hood.

Authors' addresses: Vidya A. Chhabria, University of Minnesota, USA; Vipul Ahuja, Qualcomm Technologies Inc., India; Ashwath Prabhu, Qualcomm Technologies Inc., India; Nikhil Patil, Qualcomm Technologies Inc., India; Palkesh Jain, Qualcomm Technologies Inc., India; Sachin S. Sapatnekar, University of Minnesota, USA.

Prior acceleration methods such as [22] trade off accuracy for speed by increasing the coarseness of element discretization. However, the advent of machine learning (ML) has presented fast and accurate solutions to these problems [15, 18, 19, 23, 30, 37, 38, 42], trained on simulation data from systems of millions of nodes.

For thermal analysis, existing ML-based solutions primarily focus on coarser system-level thermal modeling [18, 19, 29, 42]. The work in [37] predicts temperature at a finer granularity but uses coarse temperature estimates as an input. The methods in [18, 29] use post-silicon performance metrics as inputs to predict the full-chip temperature and are not suitable for predicting temperature during the design process.

For PDN analysis, the works in [15, 23] address incremental analysis and are not intended for full-chip estimation. The work in [10] addresses the vectored IR drop problem and uses ML-based IR drop analysis techniques to recommend a small subset of vectors that exercises worst-case scenarios for dynamic IR drop. The work in [38] proposes a convolutional neural network (CNN)-based implementation for full-chip IR drop prediction, using cell-level power maps as features. However, it assumes similar resistance from each cell to the power pads, which may not be valid for practical power grids with irregular grid density. The ML techniques in [37, 38] both divide the chip into regions (*tiles*), and the CNNs operate on each tile and its near neighbors. Selecting an appropriate tile and window size is nontrivial – small windows could violate the principle of locality [11], causing inaccuracies, while large windows could result in large models with significant runtimes for training and inference. Several algorithms [10, 38] create simplifications that do not account for PDN topologies and power bump locations that are vital for accurate IR drop prediction: as shown in Section 5.7, this can lead to significant inaccuracies. Our approach bypasses window size selection by providing the entire power map as input, allowing ML to learn the window size for accurate estimation of temperature and IR drop.

Our work also considers the use of ML for EM hotspot detection in PDNs. There has been little prior work in this area. The work in [17] uses a GAN to predict transient stress profiles for multisegment interconnect tree topologies by using images of the tree topologies and their current densities as input. While it is not directly comparable with our work, since it uses multiphysics models as against the empirical models in our work, it is instructive to perform a qualitative comparison. In [17], the images are of a fixed 256μm $\times$256μm size and represent only the small region of a larger chip and interconnect structure. This approach has three limitations. First, the model operates on small regions (256μm $\times$256μm) of the chip individually, i.e., if an interconnect belongs to multiple regions, the models predicts stress locally in the region unaware of the rest of the interconnect topology in the neighboring regions. The predicted stress profiles across regions may not maintain continuity across region boundaries when the parts of the interconnect in different regions are stitched back together. Second, PDNs in advanced technology nodes are dense and require a much smaller region size to represent the PDN topologies within the image. However, smaller regions sizes result in scalability and larger inaccuracy issues during stitching. Third, to predict the stress profile, the work uses current density per PDN segment as an input feature. It is computationally expensive to find the current densities in each segment as it requires a solution to the system of equations $G\mathbf{V} = \mathbf{J}$. While an ML approach can be used to solve this system of equations rapidly to obtain voltages, this may still be an a linear time solution as it would require iterating through all nodes to generate current densities on a per-segment basis. In addition, most ML methods that solve these equations for IR drop predict IR drop maps at a pixel level, rather than the voltage at each node that is essential for EM analysis at the PDN segment level.

Our approach overcomes the above limitations by performing EM hotspot classification using a single inference on the entire chip, instead of smaller regions. It encapsulates the estimation of current densities on a per-segment basis into a one-time training step and uses the on-chip power, PDN density distributions, power pad locations, and temperature as inputs to directly predict EM-prone segments during inference.

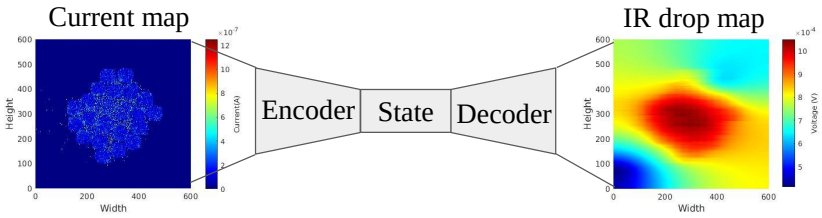

Fig. 1. Image-to-image translation using EDGe network.

1.2 Overview of our approach

In this work, we translate the static analysis problems of thermal, IR drop, and EM hotspot classification into image-to-image translation tasks, and the corresponding dynamic analysis problems into sequence-to-sequence translation tasks. We employ fully convolutional (FC) EDGe networks for rapid and accurate thermal and PDN analysis for ML image and sequence-based translation tasks. FC EDGe networks have proven to be very successful with image-related problems with 2D spatially distributed data [3, 25, 28, 31] when compared to other networks that operate without spatial correlation awareness. For transient thermal analysis where the time constant of changes is large, we use long-short-term-memory (LSTM) based networks that maintain a memory of analyses at previous time steps. For transient IR drop, we employ a U-Net based model to capture high-frequency waveforms. Based on these concepts, we propose three novel ML-based analyzers:
ThermEDGe: *Full-chip* static and transient thermal analysis

- Inputs: Power distributions
- Outputs: Temperature distributions

IREDGe: *Full-chip* static and transient IR drop analysis

- Inputs: Power distributions, PDN density map, and power bump patterns
- Outputs: IR drop distributions

EMEDGe: EM hotspot classification

- Inputs: Power distributions, PDN density map, power bump patterns, and temperature distributions
- Outputs: Per PDN layer EM-prone PDN segments

The predicted output temperature and IR drop contours are at the accuracy level of a million-node ground truth simulation. The fast inference times of these methods enable **full-chip** thermal, IR drop analysis, and EM hotspot classification in milliseconds, as opposed to several hours using commercial tools. We obtain average errors of 0.6%, 0.008%, and 8.5% (false positives) for ThermEDGe, IREDGe, and EMEDGe, respectively, over a range of testcases.

Fig. 1 shows a general structure of an EDGe network. It consists of two parts: (i) the encoder/downsampling path, which captures global features of the 2-D distributions of power dissipation, and produces a low-dimensional state space and (ii) the decoder/upsampling path, which transforms the state space into the required detailed outputs (temperature or IR drop contours). The EDGe network is well-suited for PDN/thermal analyses because:

(a) The convolutional nature of the encoder captures the dependence of both problems on the *spatial distributions of power*. Unlike CNNs, EDGe networks contain a decoder that acts as a generator to convert the extracted power and PDN density features into accurate high-dimensional full-chip temperature and IR drop contours.

(b) The trained EDGe network model for static analysis is *design-independent*: it only stores the weights of the convolutional kernel. Moreover, the same filter can be applied to *any* chip of any size for a given technology and packaging solution. The selection of the network topology (convolution filter size, number of convolution layers) is related to the expected sizes of the hotspots rather than the size of the chip: these sizes are generally similar for a given application domain, technology, and packaging choice.

(c) Unlike prior methods [17, 38] that operate tile-by-tile, where finding the right tile and window size for accurate analysis is challenging, *the choice of window size is treated as an ML hyperparameter tuning problem* to decide the amount of spatial input information.

This work represents the development and maturation of a previous conference publication [7] that addressed the static and transient thermal analysis problems and static IR drop analysis problem using encoder decoder based generative (EDGe) networks. We extend this to demonstrate the use of EDGe networks for transient IR drop analysis and the EM hotspot classification. The organization of the paper is as follows: Section 2 discusses each analysis in a traditional sense and formulates each analysis into an ML-solvable problem. Section 3 justifies the use of EDGe network architecture for these analyses. Section 4 describes golden data generation and the supervised ML model training details. Section 5 evaluates EDGe networks in terms of speed and accuracy compared to their golden solvers and prior art counterparts.

2 ADAPTING THERMAL AND PDN ANALYSIS TO ML

The core underlying computations required for PDN analysis (for both IR drop and EM) and thermal analysis are fundamentally similar, solving partial differential equations (PDE) of similar form. This section highlights the conversion of these conventional VLSI-world analyses into ML-world tasks.

2.1 Static and transient thermal analysis

The on-chip temperature distribution is governed by the parabolic PDE:

$$k_t \nabla^2 T + g(\mathbf{r}, t) = \rho c_p \frac{\partial T(\mathbf{r}, t)}{\partial t} \tag{1}$$

Here, $\mathbf{r}$ is the spatial coordinate of the point at which temperature is being analyzed, t is time (in seconds), g is the power density per unit volume (in W/m^3), c_p is the heat capacity of the chip material (in J/kg K), and ρ is the density of the chip material (in kg/m^3). Therefore, finding an on-chip temperature profile involves solving $T(r, t)$ given a power density distribution $g(r, t)$ of the chip.

Traditional techniques for solving (1) in either steady-state (i.e., with $\partial T(\mathbf{r}, t)/\partial t = 0$) or transient state are based on the finite difference method (FDM) or finite element method (FEM), which discretize the differential operator or the temperature field across space and time. In steady-state, the solution to the above PDE amounts to solving a system of linear equations of the form $G\mathbf{T} = \mathbf{P}$ [41, 44] where G is a $N \times N$ conductance matrix representing connected conductances on the grid, $\mathbf{T}$ is a $N \times 1$ vector of unknown temperatures, and $\mathbf{P}$ is a vector of the generated power density values for each element. While G is sparse, in industry-sized designs N is in the order of tens of millions. As a result, on-chip temperature evaluation is computationally expensive even for the static case.

In this work, we translate the above problem of finding $T(r, t)$ for an input $g(r, t)$ given a fixed package (k_t, ρ, c_p) into an ML-solvable problem by representing on-chip power density and temperature distribution as images. The input to (1), $g(r, t)$, is represented as a power map/image for the static problem and as a sequence of power maps/video frames

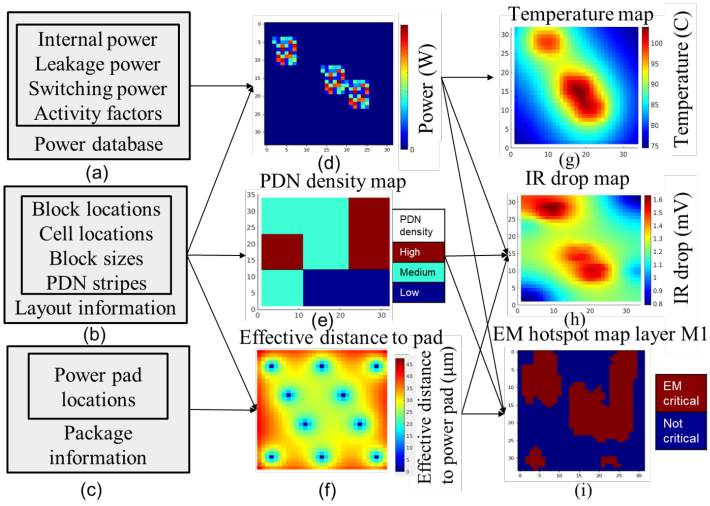

Fig. 2. Data representation: Mapping PDN and thermal analysis problems into ML-based image-to-image translations tasks.

for the dynamic problem. The maps are created by using per-instance power values from a power analysis tool and instance locations from the layout database as outlined in Fig. 2(a), (b), and (d). Each pixel in a map represents the attributes of a region on the chip. In the power map, each pixel is the sum of power values of each instance in the region and in the temperature map, each pixel is the maximum temperature of all the FDM nodes in that region. We represent the required output, $T(r,t)$ as an image (for static analysis) as shown in Fig. 2(g) or a video (for dynamic analysis). The problem can thus be converted into an image-to-image translation task for static analysis and a sequence-to-sequence translation task for dynamic analysis for a fixed k_t, c_p, and ρ.

ThermEDGe leverages ML models such as U-Nets [28] and LSTMs [16] with an encoder-decoder-like architecture for the image-to-image and sequence-to-sequence translation tasks. The computationally expensive FDM analysis is encapsulated into a one-time cost of training ThermEDGe. For a trained network, a rapid inference predicts temperature values of the chip for any given input power map. Details of the models and justification for the selection of these architectures are detailed in Section 3.

2.2 Static and transient IR drop analysis

The on-chip PDN is responsible for transmitting voltages and currents to each cell in the design. However, due to the parasitics in the PDN, a voltage drop is induced between the power pads and the cells in the design. Large voltage drops in the PDN can hurt chip performance and, in the worst case, its functionality. Consequently, it is essential to introduce checks that verify that worst-case IR drop values in the PDN are within specified limits. Simulating the PDN to calculate the IR drop in a PDN requires solving a differential equation obtained through modified nodal analysis:

$$G\mathbf{V}(t) + C\frac{\partial \mathbf{V}(t)}{\partial t} = \mathbf{J}(t) \tag{2}$$

where G is the conduction matrix, C is the diagonal capacitance matrix, and $\mathbf{V}(t)$ and $\mathbf{J}(t)$ are vectors of voltages and currents at specific instances of time. In steady-state, this reduces to $G\mathbf{V} = \mathbf{J}$, similar to its thermal counterpart.

In PDN analysis, on-chip power grid topology and package bump assignments impact IR drop. Therefore, in addition to representing $\mathbf{J}(t)$ as a 2D image or sequence of images as in the thermal analysis tasks, both static and transient IR analysis have two other inputs that must be represented as images:
(i) *A PDN density map*: This feature is generated by extracting the average PDN pitch in each region of the chip. For example, when used in conjunction with the PDN styles in [8, 35], where the chip uses regionwise uniform PDNs, the average PDN density in each region, across all metal layers, is provided as an input (Fig. 2(e)).
(ii) *An effective distance-to-power-pad*: We compute the effective distance of each instance, d_e, to N_p power pads as

$$d_e^{-1} = d_1^{-1} + d_2^{-1} + ... + d_{N_p}^{-1}$$

where d_i is the distance of the i^{th} power pad from the instance. Intuitively, the effective distance metric and the PDN density map together reflect the equivalent resistance of all paths between the instance and the pads, where the effective distance-to-power-pad serves as a proxy for resistance. Fig. 2(f) shows a typical "checkerboard" power pad layout for flip-chip packages [1, 39].

Therefore, IREDGe takes three inputs represented as images: (a) $\mathbf{J}(t)$ as power maps, (b) power pad locations that provide an effective-distance-to-power-pad map, and (c) PDN topologies as PDN density maps. A single input power map is used to predict a static IR drop across the chip (Fig. 2(h)), while a sequence of power maps in a short simulation time is used to predict the worst-case dynamic IR drop of that time period [10, 38]. The IR drop map is an image representation of the IR drop across the chip where each pixel represents the worst IR drop over all PDN nodes on the lowermost layer in that region, which is connected to the switching devices. In this way, we have mapped the static IR drop problem into an image-to-image translation task, and the dynamic IR drop problem into a sequence-to-sequence translation task.

We use U-Nets for the image-to-image translation task in static IREDGe and a modification of the U-Net that uses 3D convolutional layers for the sequence-to-sequence translation task in transient IREDGe. Unlike thermal analysis where time constants are large, IR drop consists of both high-frequency and low-frequency components. The on-chip switching activity contributes the high-frequency components while the package parasitics contribute to the low-frequency components. In our testcases, we observe that regions with high switching activity are where high-frequency components dominate and regions with low switching activity are where the low-frequency components dominate. This is highlighted in Fig. 3 for a PDN testcase in a 12nm FinFET technology. The high-frequency components contribute to the larger IR drop due to the high switching activity making it essential to capture this accurately. For sequence-to-sequence translation tasks, it has been found in ML literature that 3D convolutional layers better capture local temporal information (high frequency components) than LSTMs (which better capture global temporal information) [26, 40] . Therefore, we use a U-Net model with 3D convolutions as in [10] to predict transient IR drop. Static IR drop is predicted using a U-Net based model as in static ThermEDGe.

2.3 EM hotspot classification

EM, a phenomenon caused due to the transport of metal atoms due to the momentum of current-carrying electrons, is a major reliability concern in PDNs. Empirical EM constraints provided with commercial PDKs dictate that the current density j in each metal segment of the PDN be less that specified limits, which are governed by the Blech filter [5] and Black's [4] equation. While there has been recent interest in using accurate multiphysics model-based solutions of Korhonen's equation [21, 33, 34, 36], EM rules in today's mainstream design technologies do not provide support for these models, and therefore we use empirical models in this work.

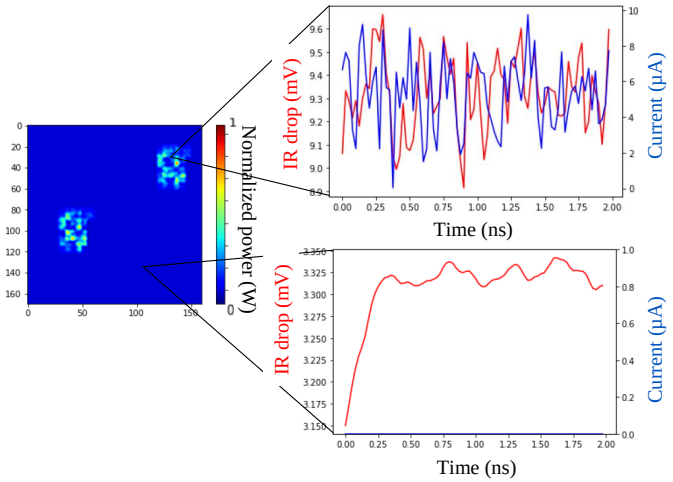

Fig. 3. High-frequency components of IR drop in regions with high switching activity and low-frequency components of IR drop in regions without activity.

In a typical cutting-edge commercial PDK, EM hotspot classification requires comparing the current density in each segment against specified EM limits, i.e., $j \leq j_{max}$, where this limit is modified for shorter wires in the spirit of the Blech criterion. The current density through each segment is obtained by solving (2). Therefore, estimating the current densities through every segment is also computationally very expensive and runtimes depend on the number of PDN nodes.

Fig. 2(i) shows the EM hotspot map for M1 of the PDN where the red regions indicate EM-critical regions, i.e., at least one PDN segment in that region is EM-critical and the blue segments are non-EM-critical regions. In contemporary technologies, even though the upper metal layers of the PDN carry large unidirectional currents, it is the lower metal layers that are more susceptible to EM failures than upper metal layers due to their small cross-sectional areas. Therefore, the sizes and number of hotspots decrease as we move to the upper metal layer of the PDN stack.

Identifying a minority class with a several class imbalance is generally a difficult problem in ML [14]. We overcome this using a variety of approaches that are used in the conventional ML literature [6, 12]. In addition, we use an aggressive EM threshold to successfully train EMEDGe to identify *EM-prone* regions and avoid misidentifying an EM-critical segment as EM-safe. The aggressive thresholds also help overcome the class-imbalance (less than 7% of PDN segments are EM-critical in M5) issue in higher metal layers that are less EM-critical, allowing for successful classification. Therefore, EMEDGe, an ML-assisted EM-prone hotspot identifier, is consistent with the industry preference where a fast ML inference can quickly identify EM-prone PDN segments, and an accurate yet fast small-scale simulation can evaluate the predicted hotspots during chip sign-off.

In this work, we propose using ML to overcome the computationally expensive nature of estimating the current density in each segment by using EMEDGe to predict EM-prone regions directly through a fast inference. We set up EMEDGe as a hotspot classifier that recommends EM-prone regions instead of identifying truly EM-critical segments directly. Since EM requires local change of the power grid density, it is helpful to flag the larger region for PDN resynthesis. In our framework, EMEDGe flags EM-prone regions that can later be analyzed for EM-criticality using a fine-grained accurate simulation of the PDN at a much smaller scale (fewer PDN nodes) than the original problem using hierarchical PDN analysis [43]. We use an aggressive EM threshold to successfully train EMEDGe to identify EM-prone

regions and avoid misidentifying an EM-critical segment as EM-safe. The aggressive thresholds also help overcome the class-imbalance (less than 7% of PDN segments are EM-critical in M5) issue in higher metal layers that are less EM-critical, allowing for successful classification. Therefore, EMEDGe, an ML-assisted EM-prone hotspot identifier, is consistent with the industry preference where a fast ML inference can quickly identify EM-prone PDN segments, and an accurate yet fast small scale simulation can evaluate the predicted hotspots during chip sign-off.

It is important to note that the voltage predicted by IREDGe cannot be directly used for estimating the current density in each segment for EM checks for two reasons: (i) IREDGe predicts the voltage drop in the lowermost layer of the PDN (M1) and is unaware of the PDN node location and segment level information in other layers. Therefore, we cannot estimate the current density through other layers in the PDN which can potentially be EM-critical, and (ii) even if we modify IREDGe to generate the voltage at each PDN node, using this method would still be slower compared to the direct use of a single ML inference as it would still require one iteration through all PDN nodes.

In standard flows, the current densities in each PDN segment are compared against EM limits to determine if a segment is EM-critical. The EM limits are specific to a technology, the BEOL stack, and are a function of the width of the PDN stripe and temperature. EMEDGe operates on a layer basis requiring one model per metal layer used in PDN. The four spatial inputs, represented as images, to EM hotspot classification are: (a) **J** as a power map, (b) PDN topology as a PDN density map, (c) power pad locations as an effective-distance-to-power-pad map, and (d) and an on-chip temperature map obtained from either the output of ThermEDGe or solving (1)[1]. Unlike static IREDGe, which predicts worst on-chip IR drop on the *lowermost layer*, EMEDGe must predict EM hotspots which highlight EM-prone PDN segments for *each layer* in the PDN. Therefore, EMEDGe consists of multiple trained models for each layer in the PDN to predict EM hotspots separately.

EMEDGe converts the EM hotspot classification problem into an image segmentation (image-to-image translation) problem where the goal is to classify each region as either EM-prone or EM-safe. We use a U-Net model to predict the EM hotspot locations. EMEDGe is metal layer-specific, i.e., for a PDN with five metal layers, we use five U-net based models to predict the EM hotspots for that specific layer. The inputs for the five models are identical but the labels are layer-specific.

Summary: We summarize the different PDN and thermal analysis problems that we address in this work in Table 1. The table highlights the differences in inputs, outputs, and the EDGe network used for each analysis. The static analyses are treated as image-to-image translation tasks and use U-Nets the transient analyses are treated as sequence-to-sequence translation tasks and use 3D U-Nets in the case of IR drop analysis and LSTMs for thermal analysis.

3 EDGE NETWORKS FOR THERMAL AND PDN ANALYSIS

3.1 U-Nets for image-to-image translation

3.1.1 Overview of U-Nets. CNNs are successful in extracting 2D spatial information for image classification and image labeling tasks, which have low-dimensional outputs (class or label). For PDN and thermal analysis tasks, the required outputs are high-dimensional distributions of IR drop/EM hotspots and temperature contours, respectively, where the dimensionality corresponds to the number of pixels of the image and the number of pixels is proportional to the size of the chip. This calls for a generator network that can translate the extracted low-dimensional input features such

[1]We use on-chip temperature as a feature for EM hotspot classification to capture (i) temperature-dependent EM limits and (ii) the exponential dependence of current densities on temperature.

Table 1. Summary of problems addressed with the corresponding EDGe network, their input features, and output labels.

Analysis type		ML task	EDGe	Inputs	Output
Thermal analysis	**Static thermal**	Image-to-image translation	U-Net	Power map	Temperature map
	Dynamic thermal	Sequence-to-sequence translation	LSTM	Time varying power maps	Time varying temperature maps
PDN analysis	**Static IR drop**	Image-to-image translation	U-Net	Power map (time varying for dynamic), power pads locations, PDN topology, and temperature (for EM only)	Static IR drop map
	Dynamic IR drop	Sequence-to-sequence translation	3D U-Net		Worst-case dynamic IR drop map
	EM hotspot	Image-to-image translation	U-Net		EM hotspot map per-layer

as (power, PDN, effective distance to pad features, etc.) from a CNN-like encoder back into high-dimensional data representing the required output.

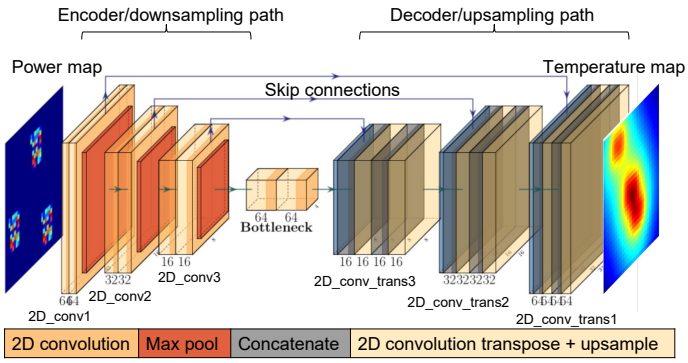

Fig. 4. U-Net-based static IREDGe, ThermEDGe, and EMEDGe.

Fig. 4 shows the structure of the EDGe network used for static PDN (IR and EM) and thermal analysis. It consists of two networks:

(a) *Encoder/Downsampling Network* Like a CNN, the network utilizes a sequence of 2D convolution and max pooling layer pairs that extract key features from the high-dimensional input feature set. The convolution operation performs a weighted sum on a sliding window across the image [13], and the max pooling layer reduces the dimension of the input data by extracting the maximum value from a sliding window across the input image. In Fig. 4, the feature dimension is halved at each stage by each layer pair, and after several such operations, an encoded, low-dimensional, compressed representation of the input data is obtained. For this reason, the encoder is also called the downsampling path: intuitively, downsampling helps understand the *"what"* (e.g., "Does the image contain power or EM or IR hotspots?") in the input image but tends to be imprecise with the *"where"* information (e.g., the precise locations of the hotspots). The latter is recovered by the decoder stages.

(b) *Decoder/Upsampling Network* Intuitively, the generative decoder is responsible for retrieving the *"where"* information that was lost during downsampling, This distinguishes an EDGe network from its CNN counterpart. The decoder is implemented using the transpose convolution [13] and upsampling layers. Upsampling layers are functionally the opposite of a pooling layer and increase the dimension of the input data matrix by replicating rows and columns.

3.1.2 Use of Skip Connections. The outputs of the PDN analysis and thermal analysis are strongly correlated to the input power – a region with high power on the chip could potentially have a IR drop or EM hotpots or temperature hotspots in its vicinity. U-Nets [28] utilize *skip* connections between the downsampling and upsampling paths, as shown in Fig. 4. These connections take information from one layer and incorporate it using a *concatenation* layer at a deeper stage skipping intermediate layers and appends it to the embedding along the z-dimension.

Skip connections combine the local embeddings from the downsampling path with the global power information from the upsampling path, allowing the underlying input features to shuttle to the layers closer to the output directly. This helps recover the fine-grained (*"where"*) details that are lost in the encoder network (as stated before) during upsampling in the decoder for detailed IR drop contours, EM hotspots, and temperature contours.

3.1.3 Receptive Fields in the Encoder and Decoder Networks. The characteristic of PDN and thermal analysis problems is that the IR drop, EM current densities, and the temperature at each location depends on both the local and global power information. The network captures local spatially correlated distributions during convolution by sliding averaging windows of an appropriate size across the input power image. For capturing the larger global impact of power on IR drop, EM current densities, and temperature max pooling layers are used after each convolution to appropriately increase the size of the *receptive field* at each stage of the network. The *receptive field* is defined as the region in the input 2D space that affects a particular pixel, and it determines the impact local, neighboring, and global features have on the analyses.

In a deep network, the value of each pixel feature is affected by all of the other pixels in the receptive field at the previous convolution stage, with the largest contributions coming from pixels near the center of the receptive field. Thus, each feature not only captures its receptive field in the input image but also gives an exponentially higher weight to the middle of that region [24]. This matches with our applications, where IR, EM, and temperature hotspots maps for a pixel are most affected by the features in the same pixel, and partially by features in nearby pixels, with decreasing importance for those that are farther away. The size of the receptive field at each stage in the network is determined by the filter sizes and the number of the convolutional and max pooling layers. The number of layers and filter sizes is determined based on the magnitude and size of the hotspot that are encountered during design iterations.

3.2 3D U-Nets for IR drop sequence-to-sequence translation

U-Nets consist of 2D convolutional layers that perform the convolution operation on the input across *all* the input channels of the input features. Due to the averaging nature of the 2D convolution operation, the resulting embedding loses fine-grained detailed information that in each channel. While this averaging effect helps capture spatial variations of power for the static IR drop and static thermal analysis problems, it fails to capture detailed fine-grained information such as the on-chip switching activity that is critical to the transient IR drop problem. Therefore, for the transient IR drop problem, we use a 3D U-Net, inspired by [10], which uses 3D convolutional layers in the encoder path instead of 2D convolutional layers. The 3D convolutional layer restricts the number of input channels on which the convolutional layer operates to a small local window based on the specified filter size. In the case of transient analysis IR drop analysis, the input channels represent time-varying power maps where each channel is the power at a specific time-step. Thus, the 3D convolutional layer works on a few time-steps of power instead of all time-steps at once. This prevents the loss of fine-grained switching information due to the averaging effect across all channels. Note that since the decoder path uses 2D convolutional layers, the interface between the 3D embedding in the encoder and 2D embedding is a sum across all channels of the embedding through the skip connections.

3.3 LSTMs for thermal sequence-to-sequence translation

Long short term memory (LSTM) based EDGe networks are a special kind of recurrent neural network (RNN) that are known to be capable of learning long term dependencies in data sequences, i.e., they have a memory component and are capable of learning from past information in the sequence.

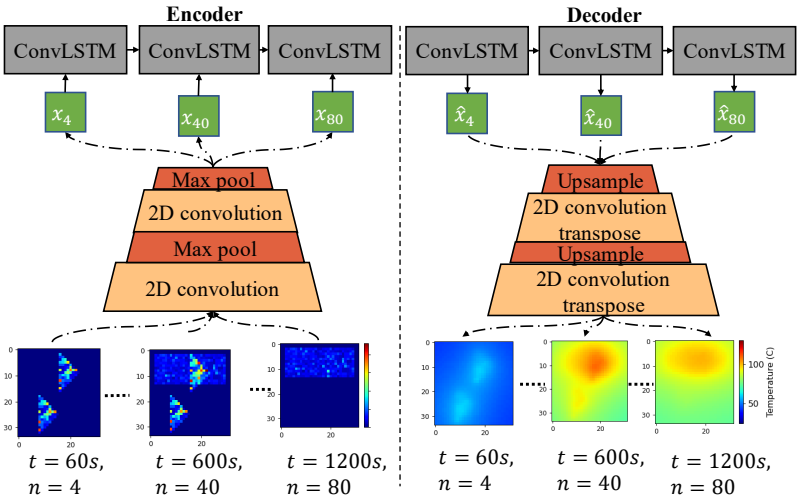

Fig. 5. LSTM-based network for transient ThermEDGe.

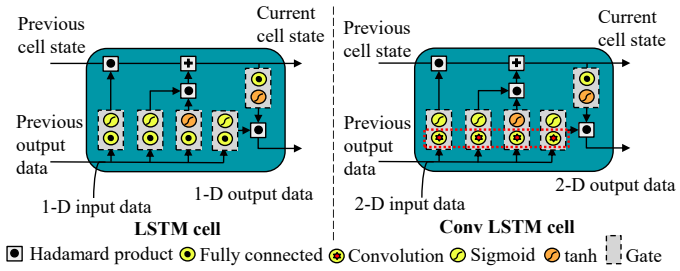

Fig. 6. Conventional (left) and ConvLSTM (right) cells.

For transient thermal analysis, the structure of ThermEDGe is shown in Fig. 5. The core architecture is an EDGe network, similar to the static analysis problem described in Section 3.1, except that the network uses additional LSTM cells to account for the time-varying component. The figure demonstrates the time-unrolled LSTM where input power frames are passed to the network one frame at a time. The LSTM cell accounts for the history of the power maps to generate the output temperature frames. The network is used for sequence-to-sequence translation in transient thermal analysis, where the input is a set of time-varying power maps and the output is a set of time-varying temperature maps (Section 2).

Similar to the static ThermEDGe network (Fig. 4), the encoder consists of convolution and max pooling layers to downsample and extract critical local and global spatial information, and the decoder consists of upsampling and

transpose convolution layers to upsample the encoded output. However, in addition, transient ThermEDGe has LSTM layers in both the encoder and decoder.

A standard LSTM cell is shown in Fig. 6 (left). While the basic LSTM cell uses fully connected layers within each gate, our application uses a variation of an LSTM cell called a convolutional LSTM (ConvLSTM) [32], shown in Fig. 6 (right). In this cell, the fully connected layers in each gate are replaced by convolution layers that capture spatial information. Thus, the LSTM-based EDGe network obtains a spatiotemporal view that enables accurate inference.

4 EDGE NETWORK TRAINING

We train ThermEDGE, IREDGe, and EMEDGe to learn temperature, IR drop contours, and EM hotspots. The training process consists of "golden" data generation and ML model training which are explained in detail in the rest of this section.

4.1 Generating Training Data

A challenge that we faced while evaluating our techniques is the dearth of public domain benchmarks that fit these applications. The IBM benchmarks [27], are potential candidates for our applications, but they assume constant currents per region and represent an older technology node. Therefore, we generate our dataset, which comprises of industry-relevant testcases, where each testcase represents industry-standard workloads for commercial designs implemented in FinFET technology.

For all three models, ThermEDGe, IREDGe, and EMEDGe, we use images/maps of size 34×32 pixels where each pixel represents a 250μm×250μm resolution tile for thermal analysis and 25μm×25μm resolution tile for PDN analysis.[2] This difference in resolution is required because thermal hot-spots have a larger spatial spread than IR hot-spots. We reiterate that although the training is performed on chips of fixed size, as we show (Section 5), inference can be performed on a chip of any size as long as the resolution (250μm×250μm or 25μm×25μm) and technology remain the same.

4.1.1 Thermal Analysis. For both the static and thermal analysis problems, we obtain our golden data from Ansys-Icepak [2] simulations. Each simulation is expensive in terms of time and memory resources: one simulation of static thermal analysis takes about 40 minutes and one simulation of transient thermal analysis for a 3000s time interval with 45 time-steps takes over 4 hours with $\approx$ 2 million nodes.

We create a 50-datapoint set for the static thermal analysis problem, where a single datapoint consists of a power map and static temperature maps. For the transient thermal analysis case, our training data consists of 150 datapoints with time-varying workloads as features and the time-varying temperatures as labels. For each testcase, we generate 45 time-step simulations that range from 0 to 3000s, with irregular time intervals from the thermal simulator. The LSTM-based network is trained using constant time steps of 15s, which enables easy integration with existing LSTM architectures, which have an implicit assumption of uniformly distributed time steps, without requiring additional features to account for the time.

4.1.2 IR drop analysis. We synthesize irregular PDNs of varying densities for each dataset element using *PDN templates*, as defined by OpeNPDN [8]. These templates are a set of PDN building blocks, spanning multiple metal layers in a 12nm commercial FinFET technology, which vary in metal utilization. For our testcases, we use three templates (high,

[2]Note that although the temperature, IR drop, and power maps work at this resolution, the ground-truth simulation consists of millions of nodes; using fewer nodes (e.g., one node per pixel) is grossly insufficient for accuracy.

medium, and low density) and divide the chip into nine regions. As outlined in Section 2.2, we use a checkerboard pattern of power pads that vary in the bump pitch and offsets across the dataset.

For static analysis, we create a 5000-datapoint set with a combination of 50 different power distributions, 10 different PDN densities, and 10 different patterns of power pad distributions. Unlike thermal analysis which has only power as the input feature, IR drop analysis as three additional input features (Refer 1) as degrees of freedom requiring a larger dataset to successfully train the model. For both the static and transient PDNs and power maps, we use an open-source modified nodal analysis-based IR drop solver PDNSim [9] to generate "golden" datapoints[3]. A single static IR drop simulation for a PDN with 2M nodes takes 20 minutes, and a single transient IR drop simulation for a 2ns duration takes ≈40 minutes. The simulation generates the voltage drop at every PDN node. We use the voltages to create an IR drop map represented at the same resolution as the input power map, where each pixel represents the maximum IR drop of all PDN nodes in a 25μm×25μm region.

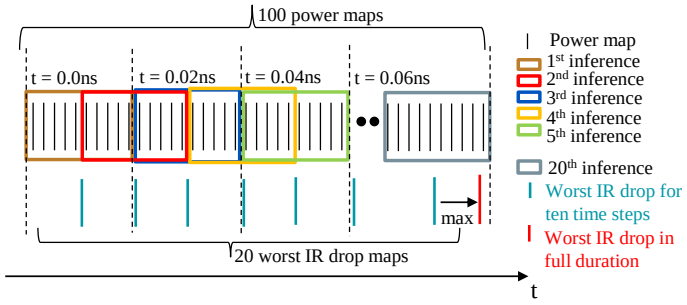

Fig. 7. Transient IREDGe power map sequence used for training and inference. The U-Net based model with 3D convolutions predicts IR drop for each inference.

Due to the dearth of modern public domain transient PDN analysis benchmarks, we use the benchmarks for static analysis and add randomly generated time-varying components to the power-consuming areas of the chip to generate our transient benchmarks. These components model the switching activity within the current power mode to generate time-varying power distributions. We generate 100 time-step power maps where each time step represents one-hundredth of a 2ns clock period.

The U-Net is trained to predict the worst IR drop for an overlapping sequence of ten time-steps. Although we use a 3D convolution layers that work on a small local window of neighboring time-steps we still use an overlapping sequence of time-steps for training and inference. The overlapping sequence provides the model neighboring past and future switching activity to accurately predict the worst IR drop for the current sequence. The number of time steps and number of overlapping time steps considered during training are ML hyperparamters that are tuned for best accuracy. Fig. 7 illustrates how transient IREDGe uses the sequence of power maps. The figure shows that we predict the worst IR drop for every ten time-steps in the sequence of power maps but use an overlapping five-time-step sequence of power maps. Therefore, to predict the IR drop for a t = 2ns duration, we perform 20 inferences. We find that working at this finer granularity with multiple inferences, where each inference predicts the worst IR drop for fraction of the clock period, provides better accuracy than using all the 100 time-steps of the input power maps with a single inference.

[3]The open-source version of PDNSim is for static IR drop analysis only. Therefore, we modify the source code to perform dynamic IR drop analysis as in (2).

4.1.3 EM hotspot classification. While PDNSim [9] generates the voltage at every PDN node, the “golden" data for training EMEDGe requires estimating the current in each segment and checking it against the PDK-specified EM limits for that segment. We use PDNSim to generate the current densities through each segment given power distribution, power pad bump distribution, and temperature distributions for each testcase and compare against the technology-specific EM limits. The limits are set based on process design kit (PDK) requirements for a commercial 12nm FinFET technology. The PDK specifies limits as a function of lifetime, temperature, layer, and length for a given width. In our work, we use a 10 year lifetime and derate the limit for each PDN segment based on the input temperature using the limit characterization from the PDK and flag each segment in the PDN as EM-critical or not. We convert the per-segment information into an EM hotspot map, which has the same resolution as the input images, by annotating a pixel as critical if any PDN segment in the region represented by the pixel has an EM-critical segment. The runtimes for generating a single data point in the training dataset for EMEDGe are identical to those of the static IREDGe simulations.

Similar to IREDGe, our training datasets for EMEDGe consists of 5000 datapoints for each of the five models (for the five PDN layers) which is generated by a combination of 50 different power distributions, 10 different PDN densities, 10 different patterns of power pad distributions, and different temperature combinations. Unlike IR drop analysis, where we train a single model to predict the on-chip IR drop, for the EM hotspot classification problem, we predict EM hotspots for every layer in the PDN. For the PDNs that we synthesize, which have five layers in the PDN stack, we train five U-Nets to predict EM hotspots in each specific layer.

In our testcases, the metal layers M1 and M2 have 30-50% of their PDN segments as EM-critical, M5 has less than 7% of EM-critical segments, and M8 and M9 have zero EM-critical segments in the dataset. As stated in Section 2.3, the small number of EM-critical segments in M5 makes it extremely challenging to capture hotspots in this layer due to a severe imbalance in data. This imbalance is addressed by using the aggressive threshold across all layers such that the fraction of the number of EM hotspots in M5 increases from 7 to 15%. In this way 15% of M5 is *EM-prone* while 7% is EM-critical. While using an aggressive threshold does increase the number of false positives on the original threshold, it ensures EMEDGe does not miss reporting any EM-prone regions. The flagged regions can be checked with a more accurate detailed analysis on a much smaller scale (a few thousand nodes as compared to millions) than the original problem itself.

4.2 Model Training

We split the data in each set for training, validation, and test sets for each of the models that we train. The training dataset is normalized by subtracting the mean and dividing by the standard deviation and is used to train the network using an ADAM optimizer [20] where the loss function is a pixel-wise mean square error (MSE). The convolutional operation in the encoder and the transpose convolution in the decoder are each followed by ReLU activation to add non-linearity and L2 regularization to prevent over fitting. The hyperparameters used for training the models are shown in Table 2.

The general architecture for the models used in static ThermEDGe, IREDGe, and EMEDGe is U-Net based and is highlighted in Fig. 4, and architecture of transient ThermEDGe is highlighted in Fig. 5. While the general architecture between all the U-Net-based models is similar, there are some differences. The key differences lie in the convolution layer type (2D vs. 3D), number of layers, and filter sizes because the size of the hotspots (IR drop, or EM, or temperature) are different. These differences between these models are listed in Table 3. The table also lists the number of trainable parameters in each model. It is important to note that the number of parameters in the model is independent of the size of the chip but scales with the size of the hotspot, which is generally similar for a given application domain, technology,

Table 2. ThermEDGe, IREDGe, and EMEDGe training hyperparameters

Training hyperparameters	Static ThermEDGe	Transient ThermEDGe	Static IREDGe	Transient IREDGe	EMEDGe (all models)
Loss function	Pixelwise MSE				Cross Entropy Loss
Learning rate	1.00E-03			1.00E-04	
Decay rate	0.98				
Decay steps	1000				
Regularizer	L2				
Regularization rate	1.00E-05			1.00E-06	
Optimizer	ADAM				
Epochs	500	200	500	200	200

Table 3. ThermEDGe, IREDGe, and EMEDGe ML model parameters

Layer hyperparameters		Static ThermEDGe	Transient ThermEDGe	Static IREDGe	Transient IREDGe	EMEDGe (all models)
conv1	filter size	5×5	5×5	3×3	$3 \times 3 \times 3$	3×3
conv_trans1	# filters	64	64	64	32	32
conv2	filter size	3×3	3×3	3×3	$3 \times 3 \times 3$	3×3
conv_trans1	# filters	32	32	32	32	32
conv3	filter size	3×3	-	3×3	$3 \times 3 \times 3$	3×3
conv_trans3	# filters	16	-	16	64	64
conv4	filter size	-	-	-	$3 \times 3 \times 3$	3×3
conv_trans4	# filters	-	-	-	128	128
ConvLSTM	filter size	-	7×7	-	-	-
	# filters	-	16	-	-	-
Max pool layer filters		2×2	2×2	2×2	2×2	2×2
#Trainable parameters		132,769	235,521	133,921	458,337	261,089

and packaging choice. A change in hotspot size or resolution demands a change in the number of layers and receptive field sizes of the models to accurately capture temperature, IR drop, and EM hotspots.

The models are trained in Tensorflow 2.1 on an NVIDIA GeForce RTX2080Ti GPU. Training runtimes are as follows: 30m each for static ThermEDGe and IREDGe, 6.5h for transient ThermEDGe, 5 hours for transient IREDGe, and 75 mins for each U-Net model in EMEDGe. We reiterate that this is a one-time cost for a given technology node and package which is amortized over repeated use over many design iterations for multiple chips.

5 EVALUATION OF EDGE NETWORKS

5.1 Experimental Setup and Metric Definitions

ThermEDGe, IREDGe, and EMEDGe are implemented using Python3.7 within a PyTorch 1.6 framework. We test the performance of our models on 40 datapoints reserved in the testset (Section 4.2), labeled T1–T40. As mentioned in Section 4, due to the unavailability of new, public domain benchmarks to evaluate our experiments, we use benchmarks that represent commercial industry design workloads.

Error Metrics As a measure of goodness of ThermEDGe and IREDGe predictions, we define a discretized regionwise error, $T_{err} = \left|T_{true} - T_{pred}\right|$, where T_{true} comes from ground truth image, generated by commercial tools, and T_{pred} from the predicted image, generated by ThermEDGe; IR_{err} is computed similarly. We report the average and maximum values of T_{err} and IR_{err} for each testcase. In addition, the percentage mean and the maximum error is listed as a fraction of a temperature corner, i.e., 105°C for thermal analysis and as a fraction of VDD= 0.7V for IR drop analysis.

Table 4. Results of ThermEDGe across 10 testcases.

Static ThermEDGe			Transient ThermEDGe		
Test	**Avg. T_{err} (C)**	**Max T_{err} (C)**	**Test**	**Avg. T_{err} (C)**	**Max T_{err} (C)**
T1	0.64 (0.61%)	2.76 (2.63%)	T6	0.51 (0.49%)	5.59 (5.32%)
T2	0.63 (0.60%)	2.67 (2.54%)	T7	0.58 (0.55%)	6.17 (5.88%)
T3	0.65 (0.62%)	2.93 (2.79%)	T8	0.57 (0.54%)	5.83 (5.55%)
T4	0.48 (0.46%)	2.22 (2.11%)	T9	0.52 (0.50%)	6.32 (6.02%)
T5	0.75 (0.71%)	2.86 (2.72%)	T10	0.56 (0.53%)	7.14 (6.80%)

To evaluate EMEDGe, we use standard binary classification metrics such as accuracy, F1 scores, and area under the ROC curve (AUROC).

5.2 Static ThermEDGe Accuracy

A comparison between the commercial tool-generated temperature and the ThermEDGe-generated temperature map for T1–T5 are listed in Table 4. The runtime of static ThermEDGe for each of the five testcases of size 34×32, is approximately 1.1ms in our environment. Across the five testcases (five rows of the table), ThermEDGe has an average T_{err} of 0.63°C and a maximum T_{err} of 2.93°C. These numbers are a small fraction of the maximum ground truth temperature of these testcases (85 – 150°C). The fast runtimes imply that our method can be used in the the inner loop of a thermal optimizer, e.g., to evaluate various chip configurations under the same packaging solution (typically chosen early in the design process). For such applications, this level of error is very acceptable.

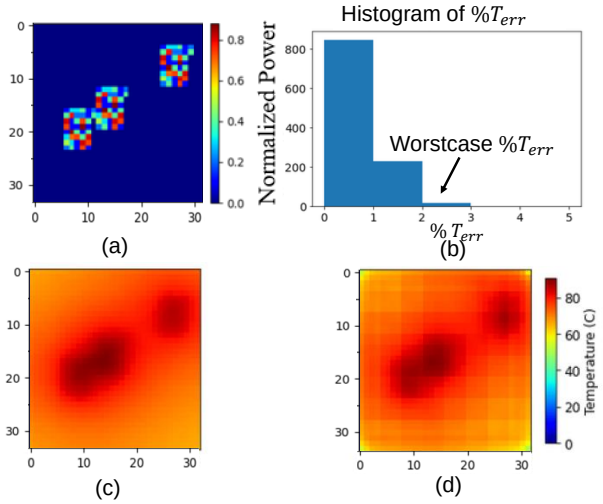

Fig. 8. Static ThermEDGe temperature estimation on T1: (a) input normalized power distribution, (b) histogram of T_{err}, (c) ground truth, and (d) predicted temperature map.

A graphical view of the predicted map for T1 is depicted in Fig. 8. For a given input power distribution (Fig. 8(a)), we compare the true temperature in Fig. 8(c) against the ThermEDGe-generated temperature contours plots in Fig. 8(d). The discrepancy is visually seen to be small. Numerically, the histogram in Fig. 8(b) shows the distribution of T_{err} where the worst-case T_{err} is 2.63% of the temperature corner.

5.3 Transient ThermEDGe Accuracy

Transient ThermEDGe predicts the output 200-frame temperature sequence at a 15s interval for the input power sequence. We summarize the results in Table 4. Across the five testcases, the prediction has an average T_{err} of 0.52% and a maximum T_{err} of 6.80% as shown. The maximum T_{err} in our testcases occur during transients that do not have long-lasting effects (e.g., on IC reliability). The magnitude of the maximum T_{err} at sustained peak temperatures is much lower, and is similar to the average T_{err}. The inference runtime to generate a 200-frame temperature contour sequence takes 10ms in our setup. In light of these millisecond runtimes these small errors are negligible.

Fig. 9 (left) shows a single frame of a video representing time-varying power maps for T6, where each frame (time-step) is after a 15s time interval. The corresponding ground truth and predicted temperature contours at the current time step are shown in the center and right, respectively, of the figure.

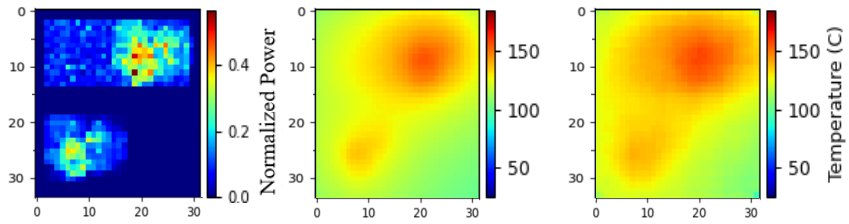

Fig. 9. Transient ThermEDGe results on T6 highlighting the input power map (left), commercial tool (center) temperature contours, and transient ThermEDGe predicted IR drop contours at a specific time-step

5.4 Static IREDGe Accuracy

We compare IREDGe-generated contours against the contours generated by PDNSim [9]. Across the five testcases in Table 5, IREDGe has an average IR_{err} of 0.053mV and a worst-case max IR_{err} of 0.34mV which corresponds to 0.008% and 0.048% of VDD respectively. Given that static IR drop constraints are 1–2.5% of VDD, a worst-case error of 0.34mV is acceptable in light of rapid runtimes. We list the results of the testcases in Table 5 where the percentage errors in IR_{err} are listed as fraction of VDD= 0.7V.

Table 5. Results of static IREDGe for 10 different testcases. T16-T20 have a chip size that was not in the training set.

Chip size: 34x32			**Chip size: 68x32**		
	Avg. IR_{err} **(mV)**	**Max** IR_{err} **(mV)**		**Avg.** IR_{err} **(mV)**	**Max** IR_{err} **(mV)**
T11	0.052 (0.007%)	0.26 (0.03%)	T16	0.035 (0.005%)	0.16 (0.02%)
T12	0.074 (0.011%)	0.34 (0.05%)	T17	0.054 (0.008%)	0.42 (0.06%)
T13	0.036 (0.005%)	0.21 (0.03%)	T18	0.035 (0.005%)	0.35 (0.05%)
T14	0.053 (0.008%)	0.24 (0.03%)	T19	0.068 (0.010%)	0.22 (0.03%)
T15	0.051 (0.007%)	0.23 (0.03%)	T20	0.061 (0.009%)	0.38 (0.05%)

A detailed view of T11 is shown in Fig. 10. It compares the IREDGe-generated IR drop contour plots against contour plot generated by [9]. The input power maps, PDN density maps, and effective distance to power pad maps are shown in Fig. 10(a), (b), and (c) respectively. Fig. 10(d) and (e) shows the comparison between ground truth and predicted value

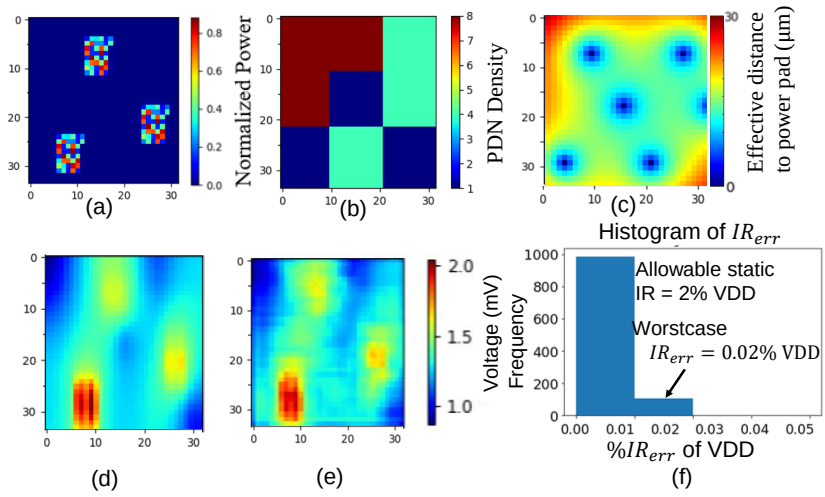

Fig. 10. Static IREDGe data for T11: input (a) power map, (b) PDN density map, (c) effective distance map, output (d) ground truth, (e) predicted IR drop map, and (f) histogram of IR_{err}.

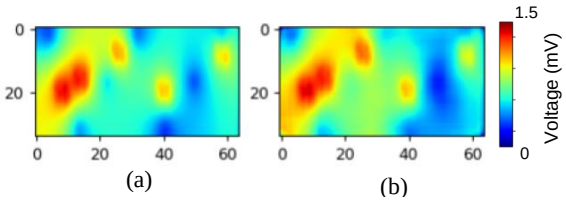

Fig. 11. Comparison between actual (left) and IREDGe-predicted (right) IR drop contours for images of size 68×32 using a model that was trained on images of size 34×32.

Table 6. Results of transient IREDGe on testcases T21-T25.

Test	**Avg.** IR_{err} **(mV)**	**Max** IR_{err} **(mV)**
T21	0.47 (0.07%)	2.46 (0.35%)
T22	0.43 (0.06%)	2.24 (0.32%)
T23	0.44 (0.06%)	2.15 (0.31%)
T24	0.3 (0.04%)	2.36 (0.34%)
T25	0.58 (0.08%)	2.03 (0.29%)

for the corresponding inputs. It is evident that the plots are similar; numerically, the histogram in Fig. 10(f) shows the IR_{err} where the worst IR_{err} is less than 0.02% of VDD.

Size-independence: Since the EDGe models only comprise the trained weights of the convolutional kernels, the same model can be reused to predict IR drop or temperature contours of a chip of a different size. We demonstrate this using IREDGe on chips of a different size (T16 – T20), using an input power distribution of size 68×32. Fig. 11 compares the actual IR drop of T16 (left) and the IREDGe-predicted (right) using a model which was trained on 34×32 power maps. The results on T16 – T20 are in Table 5.

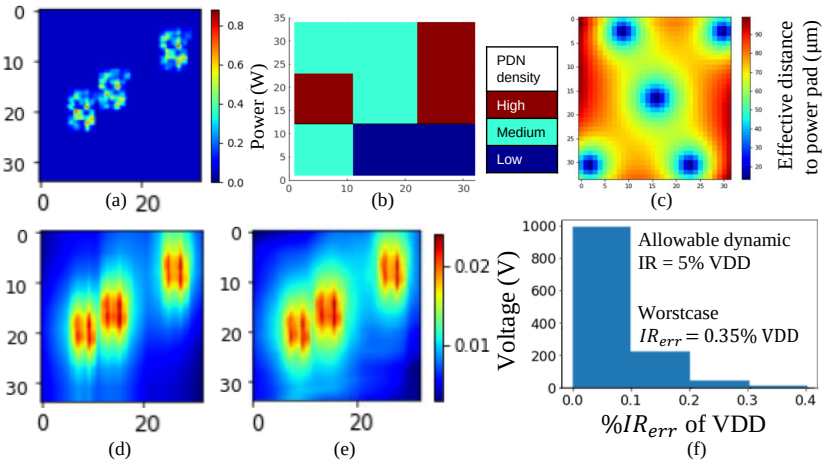

Fig. 12. Transient IREDGe data for T21: input (a) power map at time t = 1ns, (b) PDN density map, (c) effective distance map, output (d) ground truth, (e) predicted worst-case dynamic IR drop map for time period 0 to 2ns, and (f) histogram of IR_{err}.

5.5 Transient IREDGe Accuracy

We compare transient IREDGe against the IR drop contours generated by the ground-truth solver. Table 6 shows the results of five testcases. The average IR_{err} and max IR_{err} is 0.58mV and 3.03mV which corresponds to 0.08% and 0.43% of VDD respectively. Compared to industry-standard transient IR drop limits which can be as high as 5–10% of VDD, these errors are minimal.

A detailed result for T21 is shown in Fig. 12. The inputs including the power map (at t = 1ns), the PDN density map, and the effective-distance-to-power-pads map, are shown in Fig. 12(a), (b), and (c) respectively. The ground-truth worst IR drop solution for the entire 2ns period is shown in Fig. 12(d) and is compared against the IREDGe predicted solution (maximum of the 20 individual inferences as shown by the red line in Fig. 7) in Fig. 12(e). Visually, the predicted IR contours look similar to the ground-truth contours capturing the three hotspots. The histogram in Fig. 12(f) represents the IR_{err} as a fraction of VDD for all pixels in the IR drop map. This testcases has a worst-case IR_{err} of 0.35% of VDD with an IR drop limit of 5% of VDD.

5.6 IREDGe Compared with PowerNet

We implemented a version of PowerNet [38] for both static and transient IR drop analysis. For both analysis, the layout is divided into tiles, and the CNN features are the 2-D power distributions (toggle rate-scaled switching and internal power, total power, and leakage power) within each tile and in a fixed window of surrounding tiles. The trained CNN predicts the IR drop on a tile-by-tile basis by sliding a window across all tiles on the chip. The work uses a tile size of 5μm×5μm and takes into consideration a 31×31 tiled neighborhood (window) power information as features. For a fair comparison, we train IREDGe under a fixed PDN density and fixed power pad locations used to train PowerNet. For transient analysis, PowerNet uses a maximum CNN structure which takes the power map at the current time-step and uses it to predict a single value of IR drop for a specified tile at the current time-step. To generate the wortcase IR drop map, the maximum CNN structure finds the maximum IR drop of all time-steps for each tile on the chip. For static analysus, PowerNet leverages a CNN and predicts a single IR drop value per tile.

Qualitatively, IREDGe is superior to PowerNet in three aspects:
(i) *Tile and Window Size Selection:* It is stated in [38] that when the size of the tile is increased from 1μm×1μm to 5μm×5μm and the size of the resulting window is increased to represent 31×31 window of 25μm^2 tiles instead of 1μm^2 tiles, the accuracy of the PowerNet model improves. In general, this is the expected behavior with an IR analysis problem where the accuracy increases as more global information is available, until a certain radius after which the principle of locality holds [11]. IREDGe bypasses this tile-size selection problem entirely by providing the entire power map as input to IREDGe and allowing the network to learn the window size needed for accurate IR estimation.
(ii) *Runtimes and accuracy:* We compare IREDGe against our implementation of PowerNet on T26–35. The first five of these ten testcases are for static IR drop, while the latter five are for transient IR drop. These testcases have identical power distributions in T11–15 and T21–25 except that all the ten testscases have identical uniform PDNs and identical power pad distributions, as required by PowerNet; IREDGe does not require this.

Static IREDGe requires a *single* inference, irrespective of the size of the chip, while the static version of PowerNet performs an inference for every tile in the chip as it predicts IR drop by sliding a window on a tile-by-tile basis. In our experimental setup, it takes 75 minutes to train PowerNet, as against 30 minutes for IREDGe. Fig. 13 shows the comparison of inference times between PowerNet and IREDGe at similar error levels. At the 25μm^2 tile size, both IREDGe and PowerNet have similar accuracies for T26–30 as shown in Fig. 13(a). At this error level, for 0.68 mm^2 designs, IREDGe is 2.9× faster than PowerNet (Fig. 13(b)).

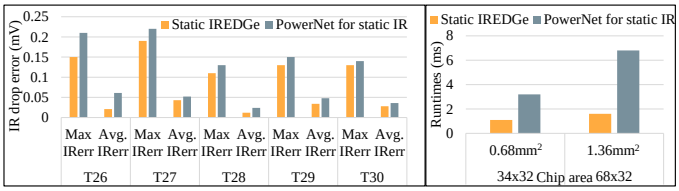

Fig. 13. Static IREDGe versus PowerNet: (a) IR drop error and (b) runtimes. IREDGe is 2.9× faster at iso-error across T26–T30 (0.68mm^2 area).

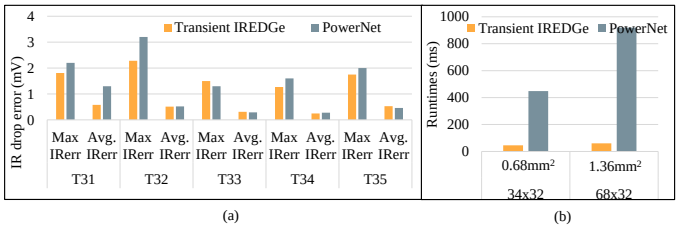

Fig. 14. Transient IREDGe versus PowerNet: (a) IR drop error and (b) runtimes. IREDGe is 9.7× (0.68mm^2) faster at iso-error across T31–T35.

The inference time for transient IREDGe is higher than static IREDGe as it requires twenty inferences with 3D convolutional layers and takes ten power maps at different time steps as input compared to static IREDGe that uses 2D convolutional layers with a single power map as input. However, transient IREDGe is still faster than PowerNet as it requires a single inference in space and 20 inferences (Refer Section 4) in time while PowerNet requires 34 × 32 inferences across space and 100 across time for the 100 time steps in a 0.2ns clock period. IREDGe can operate at a coarser granularity due to the 3D convolutional layers that capture time-dependent effects, unlike the CNN in PowerNet

that performs an inference for each time step. The errors of transient IREDGe are compared against PowerNet and shows in Fig. 14(a), and the runtime comparisons for transient IR drop are listed in Fig. 14(b). Similar to static IREDGe, for iso-accuracy (Fig. 14(a)) has transient IREDGe is 9.7× faster than PowerNet for a chip area of 0.68mm^2 and 15.3× faster for a chip of area 1.36mm^2.

(iii) *Pixelated IR drop maps:* Since PowerNet uses a CNN to predict IR drop on a tile-by-tile basis, where each tile is 5μm × 5μm, the resulting IR drop image is pixelated, and the predicted region value does not correlate well with the neighboring regions. This is highlighted in Fig. 15 which compares IR drop contours from a golden solver (Fig. 15(a)), IREDGe (Fig. 15(b)), and our implementation of PowerNet (Fig. 15(c)) for T26.

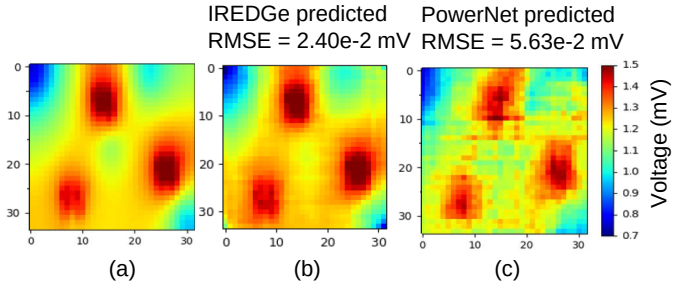

Fig. 15. IR drop comparisons on T26: (a) ground truth, (b) static IREDGe, and (c) our implementation of static PowerNet.

5.7 Impact of C4 bump locations and PDN density on IR drop

In addition to the on-chip power distributions, the locations of the C4 bumps and PDN densities dictate both static and transient IR drop distributions. The voltage source locations and PDN density impact the equivalent resistance between the on-chip instance and the power supply affecting the IR drop pattern. However, the works in [10, 38] are based on input power only and do not account for these additional factors that are critical for accurate IR drop prediction. We measure the impact these factors have on worst-case transient IR drop in Fig. 16 and highlight the importance of considering these factors as features in IREDGe. The figure shows two configurations of PDN densities and two configurations of voltage source locations for a fixed input power distribution. The worst-case transient IR drop in Fig. 16(a) is 14.2% lower than worst IR drop in Fig. 16(b) due to the higher PDN density in the former. The impact of the C4 bump distribution on IR drop is highlighted in Fig. 16(b) and (c). For the same power grid and input power distributions, the worst-case IR drop changes by 14.9% due to a change in the number and locations of the C4 bumps. Therefore, capturing these degrees of freedom is critical to accurate IR drop prediction.

Table 7 compares max IR_{err} and average IR_{err} of transient IREDGe, against a model such as [10] which does not consider the location of C4 bumps and PDN density as features. The table shows that for testcases T21–T25 these features are required for accurate IR drop prediction.

5.8 EMEDGe evaluation

As mentioned in Section 2 we convert the EM problem into an image-segmentation problem by assigning a pixel as EM-critical if any PDN segment in that region is EM-critical. Since EMEDGe operates on a region level where all the PDN segments in the region are represented by a single pixel, we lose fine-grained information at the PDN-segment level. To fairly evaluate EMEDGe we account for this inaccuracy by performing comparison against the ground-truth

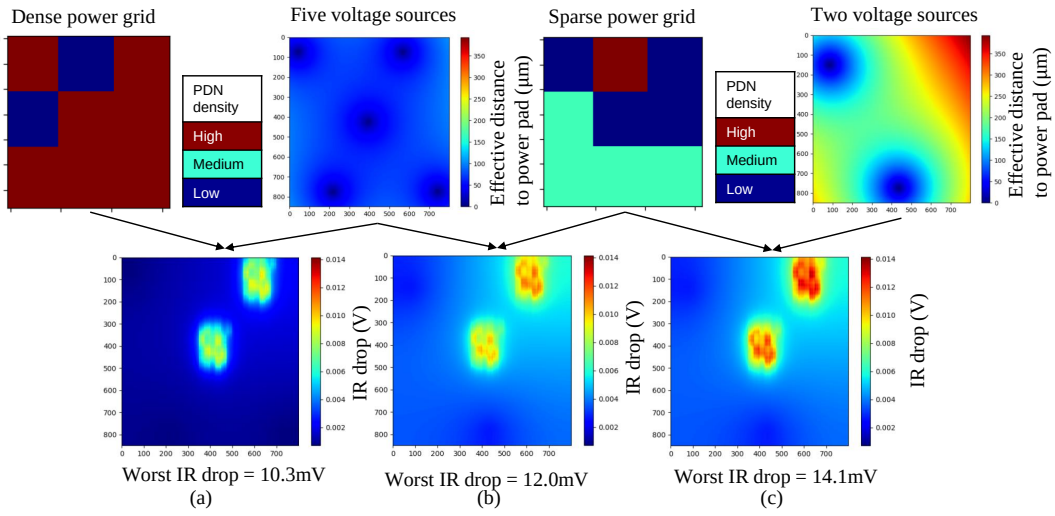

Fig. 16. Comparison of worst-case transient IR drop distributions for (a) a dense PDN with five voltage sources, (b) a sparse PDN with five voltage sources, and (c) a sparse PDN with only two voltage sources for a fixed input power map.

Table 7. Impact of C4 bump and PDN density on testcases T21-T25.

Test	Transient IREDGe (With C4 bump location and PDN density as features)		Without C4 bump locations and PDN density as features	
	Avg. IR_{err} (mV)	Max IR_{err} (mV)	Avg. IR_{err} (mV)	Max IR_{err}(mV)
T21	0.47 (0.07%)	2.46 (0.35%)	0.92 (0.13%)	4.78 (0.68%)
T22	0.43 (0.06%)	2.24 (0.32%)	1.37 (0.19%)	6.22 (0.88%)
T23	0.44 (0.06%)	2.15 (0.31%)	0.63 (0.09%)	3.15 (0.45%)
T24	0.3 (0.04%)	2.36 (0.34%)	0.74 (0.11%)	3.89 (0.55%)
T25	0.58 (0.08%)	2.03 (0.29%)	0.67 (0.10%)	3.36 (0.48%)

Table 8. EMEDGe results for binary hotspot classification with original thresholds on a per-PDN segment basis for testcases T26-T30.

Test	**M1**						**M2**						**M5**					
	Accuracy	**TP**	**TN**	**FP**	**FN**	**F1**	**Accuracy**	**TP**	**TN**	**FP**	**FN**	**F1**	**Accuracy**	**TP**	**TN**	**FP**	**FN**	**F1**
T36	0.91	116383	139666	26559	0	0.90	0.92	2031	5855	688	0	0.86	0.98	1532	31246	793	0	0.79
T37	0.89	138811	112144	31346	0	0.90	0.86	13000	15421	4534	0	0.85	0.93	1433	6397	579	0	0.83
T38	0.88	139058	108457	34840	0	0.89	0.89	5717	9313	1935	0	0.86	0.93	4471	26736	2433	0	0.79
T39	0.90	138261	115856	28248	0	0.91	0.93	5374	10472	1156	0	0.90	0.97	438	7759	214	0	0.80
T40	0.89	112828	139866	29952	0	0.88	0.92	3345	12494	1389	0	0.83	0.99	272	8078	123	0	0.82
%FP	**0.106**						**0.104**						**0.044**					

on a per-PDN-segment level. The EMEDGe-predicted EM-prone regions are converted into PDN-segment level detail by assigning all segments in the region to the same class (EM-prone or not) as the pixel.

Fig. 17 compares the EM-critical segments reported by the ground-truth solver and the EM-prone segments reported by EMEDGe for M1 and an input power map shown in Fig. 17(a). Visually, the ground truth EM hotspots from Fig. 17(c) are near-identical to the EMEDGe-predicted hotspots in Fig. 17(d). Since the EM hotspot analysis is formulated as a

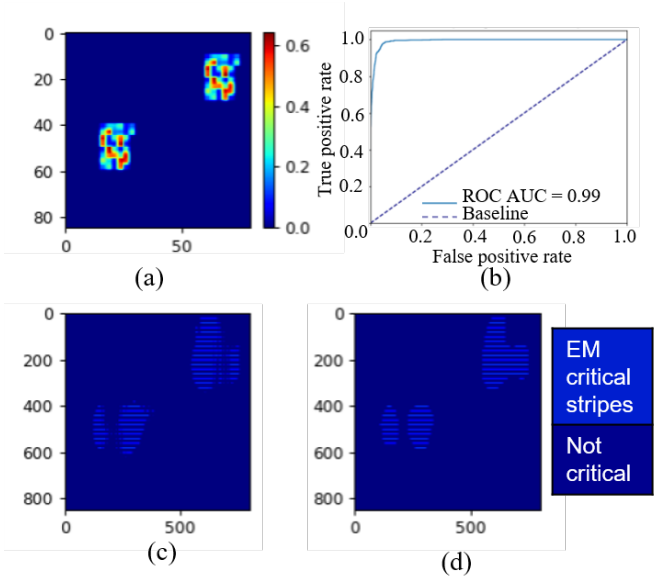

Fig. 17. EMEDGe evaluation on T36 for layer M1: input (a) input chip power map, (b) ROC curve for on a per-PDN-segment basis, (c) output ground truth EM-critical PDN segments, and (d) predicted EM-prone PDN segments.

Table 9. Runtimes of EDGe networks and golden analysis tools for a chip of area 0.68mm^2.

Analysis type	**# Nodes**	**Icepak/ PDNSim runtimes**	**EDGe network runtimes**
Static thermal	2.0 million	30 mins	1.1ms
Transient thermal	2.0 million	210 mins	10ms
Static IR drop	3âĂŞ8 million	5âĂŞ20 mins	1.1ms
Transient IR drop	3âĂŞ8 million	30âĂŞ60 mins	46ms
EM hotspot	3âĂŞ8 million	5âĂŞ20 mins	5.5ms

classical binary classification problem, we plot the receiver operating characteristic (ROC) curve, which demonstrates the performance of the model for different thresholds. The area under the ROC (AUROC) curve denotes the accuracy of the model. In this case, for the testcase T26 and the M1 layer model, we have an AUROC of 0.99 as shown in Fig. 17(b), which shows that the model has a good classification accuracy.

Even though we perform training with an aggressive threshold compared to the PDK-specified threshold we evaluate the results against PDK-specified original thresholds.

Table 8 lists the F1 score, the number of true positives, false positives, true negatives, and false negatives on a per-segment basis for the five testcases across layer M1, M2, and M5 for the scaled aggressive threshold. Even with an aggresive threshold, only the lower metal layers M1, M2, and M5 have EM hotspots, and the other layers do not have any EM-prone segments. Therefore, we do not perform any evaluation on these layers. Our findings match this industry observation where the ratio of EM-critical segments to the total number of segments in each layer reduces from the lower metal layers to the higher metal layers. Across the five testcases listed in Table 8, we obtain an average accuracy

(average of all rows) of 0.89, 0.9, and 0.96 and average F1 score of 0.90, 0.87, and 0.85 for the models corresponding to layers M1, M2, and M5, respectively, with these aggressive thresholds.

The table shows no false negatives, i.e., EMEDGe has not missed flagging out any EM-critical segment as EM-prone. While there are false positives reported, they are still small in number (≈10% of the total segments in the PDN). The flagged EM-prone segments (TP and FP) can be further evaluated in detail using a more accurate, fine grained simulation. The small number of flagged segments implies that such a simulation involves a small subcircuit, and can be fast.

5.9 Inference Runtime Analysis

Table 9 compares the inference runtimes of our ML-based EDGe network approach against the "golden" solvers. The runtimes are reported on an NVIDIA GeForce RTX 2080Ti GPU. With the millisecond inference times and the transferable nature of our trained models, the one-time cost of training the EDGe networks is quickly amortized over multiple uses within a design cycle and over multiple designs.

6 CONCLUSION

This paper addresses the compute-intensive tasks of thermal and PDN analysis by proposing the use EDGe networks as apt ML-based solutions. We successfully evaluate EDGe networks for these applications by developing two ML-based tools (i) ThermEDGe, (ii) IREDGe, and (iii) EMEDGe for rapid on-chip thermal, PDN and EM hotspot classification, respectively. The EDGe-based techniques provide millisecond-inference runtimes, and are demonstrated to be accurate when compared to ground-truth solvers, which take several hours.

REFERENCES

[1] B. W. Amick, C. R. Gauthier, and D. Liu. 2002. Macro-Modeling Concepts for the Chip Electrical Interface. In *Proc. DAC*. ACM, New York, NY, USA, 391–394.

[2] Ansys. 2018. Icepak. Retrieved Oct. 23, 2021 from https://www.ansys.com/products/electronics/ansys-icepak

[3] V. Badrinarayanan, A. Kendall, and R. Cipolla. 2017. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 12 (Dec. 2017), 2481–2495.

[4] J. R. Black. 1969. Electromigration failure modes in aluminum metallization for semiconductor devices. *Proc. IEEE* 57, 9 (Sept. 1969), 1587–1594.

[5] I. A. Blech. 1976. Electromigration in thin aluminum films on titanium nitride. *J. Appl. Phys.* 47, 4 (April 1976), 1203–1208.

[6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Intell. Res.* 16, 1 (June 2002), 321–357.

[7] V. A. Chhabria, V. Ahuja, A. Prabhu, N. Patil, P. Jain, and Sachin S. Sapatnekar. 2021. Thermal and IR Drop Analysis Using Convolutional Encoder-Decoder Networks. In *Proc. ASP-DAC*. ACM, New York, NY, USA, 690–696.

[8] V. A. Chhabria, A. B. Kahng, M. Kim, U. Mallappa, S. S. Sapatnekar, and B. Xu. 2020. Template-based PDN Synthesis in Floorplan and Placement Using Classifier and CNN Techniques. In *Proc. ASP-DAC*. IEEE, Piscataway, NJ, 44–49.

[9] V. A. Chhabria and S. S. Sapatnekar. 2021. PDNSim. https://github.com/The-OpenROAD-Project/OpenROAD/tree/master/src/psm.

[10] V. A. Chhabria, Y. Zhang, H. Ren, B. Keller, B. Khailany, and S. S. Sapatnekar. 2021. MAVIREC: ML-aided vectored IR-drop estimation and classification. In *Proc. DATE*. IEEE, Piscataway, NJ, 1825–1828.

[11] E. Chiprout. 2004. Fast flip-chip power grid analysis via locality and grid shells. In *Proc. ICCAD*. IEEE, Piscataway, NJ, 485–488.

[12] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie. 2019. Class-Balanced Loss Based on Effective Number of Samples. In *Proc. CVPR*. 9260–9269.

[13] V. Dumoulin and F. Visin. 2016. A guide to convolution arithmetic for deep learning. arXiv:1603.07285 [stat.ML]

[14] H. He and E. A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* 21, 9 (Sept. 2009), 1263–1284.

[15] C.-T. Ho and A. B. Kahng. 2019. IncPIRD: Fast Learning-Based Prediction of Incremental IR Drop. In *Proc. ICCAD*. IEEE, Piscataway, NJ.

[16] S. Hochreiter and J. Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.

[17] W. Jin, S. Sadiqbatcha, Z. Sun, H. Zhou, and S. X.-D. Tan. 2020. EM-GAN: Data-Driven Fast Stress Analysis for Multi-Segment Interconnects. In *Proc. ICCD*. IEEE, Piscataway, NJ, 296–303.

[18] W. Jin, S. Sadiqbatcha, J. Zhang, and S. X. . D. Tan. 2020. Full-Chip Thermal Map Estimation for Commercial Multi-Core CPUs with Generative Adversarial Learning. In *Proc. ICCAD*. IEEE, Piscataway, NJ.

[19] D. Juan, H. Zhou, D. Marculescu, and X. Li. 2012. A learning-based autoregressive model for fast transient thermal analysis of chip-multiprocessors. In *Proc. ASP-DAC*. IEEE, Piscataway, NJ, 597–602.

[20] D. Kingma and J. Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proc. ICLR*.

[21] M. A Korhonen, P. Borgesen, K. N. Tu, and C. Y. Li. 1993. Stress evolution due to electromigration in confined metal lines. *J. Appl. Phys.* 73, 8 (Aug. 1993), 3790–3799.

[22] J. N. Kozhaya, S. R. Nassif, and F. N. Najm. 2002. A multigrid-like technique for power grid analysis. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 21, 10 (Oct. 2002), 1148–1160.

[23] S.-Y. Lin, Y.-C. Fang, Y.-C. Li, Y. Liu, T. Yang, S.-C. Lin, C.-M. J. Li, and E. J.-W. Fang. 2018. IR drop prediction of ECO-revised circuits using machine learning. In *Proc. VTS*. IEEE, Piscataway, NJ.

[24] W. Luo, Y. Li, R. Urtasun, and R. Zemel. 2016. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *Adv. NeurIPS*. Curran Associates, Inc., Red Hook, NY.

[25] X.-J. Mao, C. Shen, and Y.-B. Yang. 2016. Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections. In *Adv. NeurIPS*. Curran Associates, Inc., Red Hook, NY.

[26] J. MÃďttÃďri, S. BroomÃĺ, J. Folkesson, and H. KjellstrÃűm. 2020. Interpreting video features: a comparison of 3D convolutional networks and convolutional LSTM networks. arXiv:2002.00367 [cs.CV]

[27] S. R. Nassif. 2008. Power Grid Analysis Benchmarks. In *Proc. ASP-DAC*. IEEE, Piscataway, NJ, 376–381.

[28] O. Ronneberger, P. Fischer, and T. Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proc. Int. Conf. Med. Image Comput. Comput.-Assisted Intervention*. Springer, Cham, 234–241.

[29] S. Sadiqbatcha, J. Zhang, H. Amrouch, and S. X.-D. Tan. 2021. Real-Time Full-Chip Thermal Tracking: A Post-Silicon, Machine Learning Perspective. *IEEE Trans. Comput.* (2021).

[30] S. Sadiqbatcha, H. Zhao, H. Amrouch, J. Henkel, and S. X. . Tan. 2019. Hot Spot Identification and System Parameterized Thermal Modeling for Multi-Core Processors Through Infrared Thermal Imaging. In *Proc. DATE*. IEEE, Piscataway, NJ, 48–53.

[31] E. Shelhamer, J. Long, and T. Darrell. 2017. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 4 (April 2017), 640–651.

[32] X. Shi, Z. Chen, H.. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo. 2015. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Adv. NeurIPS*. Curran Associates, Inc., Red Hook, NY.

[33] M. A. Al Shohel, V. A. Chhabria, N. Evmorfopoulos, and S. S. Sapatnekar. 2021. Analytical Modeling of Transient Electromigration Stress based on Boundary Reflections. In *Proc. ICCAD*. IEEE, Piscataway, NJ. (to appear).

[34] M. A. Al Shohel, V. A. Chhabria, and S. S. Sapatnekar. 2021. A New, Computationally Efficient "Blech Criterion" for Immortality in General Interconnects. In *Proc. DAC*. IEEE, Piscataway, NJ. (to appear; available at https://arxiv.org/abs/2105.08784).

[35] J. Singh and S. S. Sapatnekar. 2006. Partition-based algorithm for power grid design using locality. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 25, 4 (April 2006), 664–677.

[36] S. X.-D. Tan, M. Tahoori, T. Kim, S. Wang, Z. Sun, and S. Kiamehr. 2019. *VLSI systems long-term reliability – Modeling, simulation and optimization*. Springer, Boston, MA.

[37] J. Wen, S. Pan, N. Chang, W. Chuang, W. Xia, D. Zhu, A. Kumar, E. Yang, K. Srinivasan, and Y. Li. 2020. DNN-based Fast Static On-chip Thermal Solver. In *Proc. IEEE Symp. Semicond. Therm. Meas., Model. Manage.* IEEE, Piscataway, NJ, 65–75.

[38] Z. Xie, H. Ren, B. Khailany, Y. Sheng, S. Santosh, J. Hu, and Y. Chen. 2020. PowerNet: Transferable Dynamic IR Drop Estimation via Maximum Convolutional Neural Network. In *Proc. ASP-DAC*. IEEE, Piscataway, NJ, 13–18.

[39] F. Yazdani. 2018. *Foundations of Heterogeneous Integration: An Industry-Based, 2.5D/3D Pathfinding and Co-Design Approach*. Springer, Boston, MA, USA.

[40] J. You and J. Korhonen. 2019. Deep Neural Networks for No-Reference Video Quality Assessment. In *Proc. Intl. Conf. on Image. Process.* IEEE, Piscataway, NJ, 2349–2353.

[41] Y. Zhan, S. V. Kumar, and S. S. Sapatnekar. 2008. Thermally-Aware Design. *Found. Trends Electron. Des. Automat.* 2, 3 (March 2008), 255–370.

[42] K. Zhang, A. Guliani, S. Ogrenci-Memik, G. Memik, K. Yoshii, R. Sankaran, and P. Beckman. 2018. Machine Learning-Based Temperature Prediction for Runtime Thermal Management Across System Components. *IEEE Trans. Parallel Distrib. Syst.* 29, 2 (Feb. 2018), 405–419.

[43] M. Zhao, R. Panda, S. S. Sapatnekar, and D. Blaauw. 2002. Hierarchical analysis of power distribution networks. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 21 (02 2002), 159–168.

[44] Y. Zhong and M. D. F. Wong. 2005. Fast algorithms for IR drop analysis in large power grid. In *Proc. ICCAD*. IEEE, Piscataway, NJ, 351–357.