Runtime Stress Estimation for Three-dimensional IC Reliability Management Using Artificial Neural Network

HAI WANG, TAO XIAO, DARONG HUANG, LANG ZHANG, CHI ZHANG, HE TANG, and YUAN YUAN, University of Electronic Science and Technology of China, China

Heat dissipation and the related thermal-mechanical stress problems are the major obstacles in the development of the three-dimensional integrated circuit (3D IC). Reliability management techniques can be used to alleviate such problems and enhance the reliability of 3D IC. However, it is difficult to obtain the time-varying stress information at runtime, which limits the effectiveness of the reliability management. In this article, we propose a fast stress estimation method for runtime reliability management using artificial neural network (ANN). The new method builds ANN-based stress model by training offline using temperature and stress data. The ANN stress model is then used to estimate the important stress information, such as the maximum stress around each TSV, for reliability management at runtime. Since there are a variety of potential ANN structures to choose from for the ANN stress model, we analyze and test three ANN-based stress models with three major types of ANNs in this work: the normal ANN-based stress model, the ANN stress model with handcrafted feature extraction, and the convolutional neural network–(CNN) based stress model. The structures of each ANN stress model and the functions of these structures in 3D IC stress estimation are demonstrated and explained. The new runtime stress estimation method is tested using the three ANN stress models with different layer configurations. Experiments show that the new method is able to estimate important stress information at extremely fast speed with good accuracy for runtime 3D IC reliability enhancement. Although all three ANN stress models show acceptable capabilities in runtime stress estimation, the CNN-based stress model achieves the best performance considering both stress estimation accuracy and computing overhead. Comparison with traditional method reveals that the new ANN-based stress estimation method is much more accurate with a slightly larger but still very small computing overhead.

CCS Concepts: • **Hardware → 3D integrated circuits**; **Modeling and parameter extraction**; **Aging of circuits and systems**;

Additional Key Words and Phrases: Stress estimation, 3D IC, artificial neural network, reliability management

ACM Reference format:
Hai Wang, Tao Xiao, Darong Huang, Lang Zhang, Chi Zhang, He Tang, and Yuan Yuan. 2019. Runtime Stress Estimation for Three-dimensional IC Reliability Management Using Artificial Neural Network. *ACM Trans. Des. Autom. Electron. Syst.* 24, 6, Article 69 (November 2019), 29 pages.
https://doi.org/10.1145/3363185

Preliminary results of this work have been published in *Proceedings of the 2016 International Symposium on Quality Electronic Design (ISQED'16)* [41].
This research is supported in part by National Natural Science Foundation of China under grant No. 61404024, in part by the Fundamental Research Funds for the Central Universities under grant No. ZYGX2016J043, and in part by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.
Authors' address: H. Wang, T. Xiao, D. Huang, L. Zhang, C. Zhang, H. Tang, and Y. Yuan, University of Electronic Science and Technology of China, No. 4, Sec. 2, North Jianshe Rd. Chengdu, Sichuan, 610054, China; emails: wanghai@uestc.edu.cn, xiaotao94@qq.com, drhuang93@gmail.com, atiatiati@126.com, {zhangc, tanghe, yuany}@uestc.edu.cn.

1084-4309/2019/11-ART69 $15.00
https://doi.org/10.1145/3363185

1 INTRODUCTION

Three-dimensional integrated circuit (3D IC) is a promising technique to overcome the scaling problems encountered in traditional 2D IC. It provides benefits in enabling heterogeneous integration, alleviating interconnect barrier problem, and improving power performance, and so on [3, 37]. The benefits are provided by integrating IC dies vertically, and connecting different die layers by through-silicon vias (TSV) [28]. With so many advantages against traditional 2D IC, the 3D IC suffers from severe thermal induced reliability problems, due to its poor heat removing ability in the vertical dimension [19, 31] and the significant leakage power in today's integrated circuits [9, 36]. The TSV structure makes the reliability even worse: Caused by different materials used in TSV and die, thermal variations in both space and time lead to stress variations around TSV, which shortens the lifetime of 3D IC [15].

Reliability management techniques [7, 17, 34, 43] can be applied to 3D IC to enhance the thermal induced reliability, with power budget as constraint [26, 35]. However, most of the existing reliability management methods aim at lowering the average temperature of the chip and minimizing temperature variation across different parts of the 3D IC body, due to the lack of stress information.

Unfortunately, reliability management based on such simple temperature distribution optimization is insufficient for 3D IC. Due to the mismatch in coefficients of thermal expansion (CTE) between silicon and the filling material of through-silicon vias (TSV), significant thermal stresses will be generated in 3D IC, which will bring about the reliability problems such as cracking and timing violation [27, 38]. As a result, it is important to estimate the stress at runtime speed to enable accurate stress-aware reliability management for 3D IC.

Traditionally, stress in 3D IC can be estimated by finite element methods (FEM) [6, 16, 22, 25, 30, 40, 42] or approximated by analytical methods [2, 23]. However, none of them can be used for reliability management: The FEM-based methods are computationally too expensive and the analytical stress models can only be built for a fixed and uniform temperature distribution. Detailed discussions on the existing 3D IC stress estimation methods can be found later in Section 2.

In this work, an artificial neural network–(ANN) based runtime stress estimation method is developed for reliability management of 3D IC. The new method builds ANN-based stress model by training offline using sampled temperature and stress data. The ANN stress model is then used to estimate the important stress information, such as the maximum stress around each TSV, for reliability management at runtime. The major contributions of this work are summarized as follows:

- We have proposed an ANN stress model-based fast stress estimation framework that is able to estimate important stress information at runtime speed for 3D IC. With the stress information provided by the ANN stress model, more accurate stress-aware reliability management methods can be developed.
- We have analyzed three different ANN stress models based on three major types of ANNs: the normal ANN-based stress model, the ANN stress model with hand-crafted feature extraction, and the convolutional neural network–(CNN) based stress model. The structures of each ANN stress model are demonstrated and the functions of these structures in 3D IC stress estimation are analyzed and explained.
- We have tested the performance of the ANN-based fast stress estimation method. Experiments show that the new method is able to estimate important stress information at extremely fast speed with good accuracy for runtime 3D IC reliability enhancement.
- We have also compared the accuracy and computing overhead of the three ANN stress models. We conclude that although all three ANN stress models show acceptable capabilities

in runtime stress estimation, the CNN-based stress model achieves the best performance considering both stress estimation accuracy and computing overhead.
- The new ANN-based stress estimation method is compared with the traditional fast stress estimation method [23]. The experiments show that with a slightly larger computing overhead, the new method is much more accurate than the traditional method, thanks to its ability to consider temperature distribution/gradient around each TSV.

The remaining part of this article is organized as follows. In Section 2, we first review some important research in stress estimation for 3D IC. In Section 3, the traditional FEM-based stress analysis is presented, which serves as the golden 3D IC stress analysis method and provides sampling data for the new method. Then, we demonstrate the new runtime stress estimation method for 3D IC using ANN in Section 4. In this section, the motivation and framework of the new method are given first, followed by a comprehensive presentation of the three different ANN stress models. The experimental results showing the performance of the fast stress estimation method are given in Section 5. Finally, Section 6 concludes this article.

2 RELATED WORK

In this section, we briefly review some related research on 3D IC stress estimation.

Since stress has a significant impact on the reliability of 3D IC, measurement-based methods were proposed to analyze the temperature induced stress in 3D IC with the assistance from optical instruments. Stress analysis methods based on μ-Raman spectroscopy and precision wafer curvature technology were proposed in References [14, 29]. A method employing synchrotron X-ray micro-diffraction to estimate the stress in 3D IC was proposed in Reference [4]. However, these methods require additional optical instruments for the measurement process, and thus they cannot be used for runtime reliability management of 3D IC.

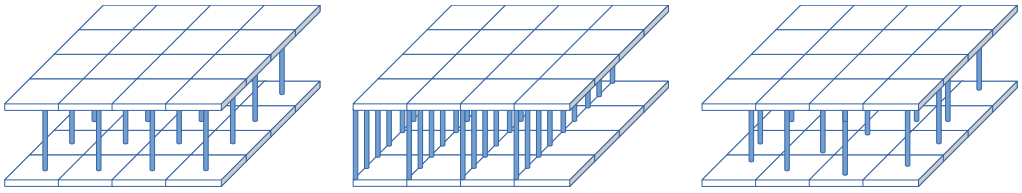
In addition, many FEM-based methods have been proposed to accurately estimate the stress around TSV [6, 22, 30]. Several faster FEM-based methods were proposed using linear superposition principle in References [16, 25, 40]. Zhou et al. improves the traditional FEM-based stress estimation method by using parallel adaptive FEM [42]. Although these methods are able to estimate the stress around TSV accurately, they cannot be applied to runtime thermal stress analysis due to the extremely high computing costs of FEM.

To speed up stress estimation, some analytical stress model-based methods have been proposed to calculate the thermal stress around TSV. Researchers proposed a uniaxial thermal stress analytical model for TSV in Reference [2]. To consider biaxial stress, an improved thermal stress model was proposed in Reference [23]. However, these analytical stress models are designed to perform corner-based analysis with a fixed and uniform temperature distribution. In other words, they are not designed to consider temperature variation in time and temperature gradient in space, which both change drastically in multi-core 3D IC systems. As a result, these analytical stress models cannot be used for runtime reliability management that requires real-time stress information according to current spatial temperature distribution on chip.

3 TRADITIONAL STRESS ANALYSIS FOR 3D ICS USING FEM-BASED METHOD

In this section, we introduce some preliminaries of FEM-based stress modeling for 3D IC with the appearance of TSVs. The FEM-based method serves as the golden 3D IC stress analysis method. It also provides sampling data for the training process of the new runtime stress estimation method.

Different 3D IC chips may have different TSV distributions. For example, the TSVs may distribute uniformly [39] on chip as shown in Figure 1(a), and they may also form nonuniform distribution [24] as shown in Figure 1(b) or random distribution as shown in Figure 1(c). For

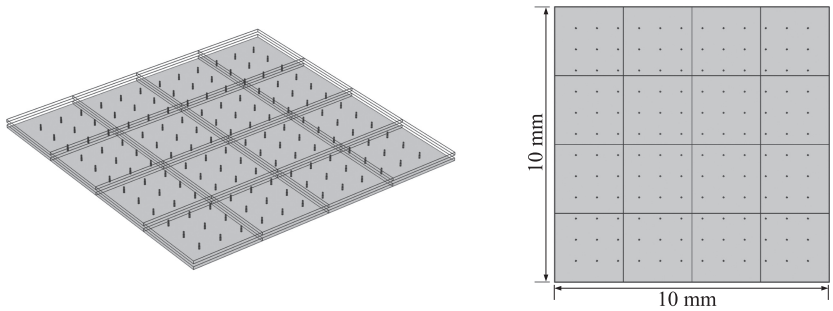


(a) A 3D IC with uniform TSV distribution.

(b) A 3D IC with nonuniform TSV distribution.

(c) A 3D IC with random TSV distribution.

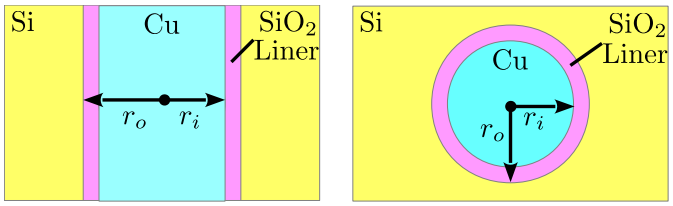
Fig. 1. Three-dimensional ICs with different TSV distributions. For simplicity, package structure is not shown in the figure.



(a) Side view showing the two die layers connected with 144 TSVs.

(b) Bottom view of chip showing the 16-core architecture for each layer.

Fig. 2. A 32-core (16 cores on each layer) 3D IC with uniform TSV distribution built in COMSOL. Package structure is not shown in the figure.



(a) Cross-section view.

(b) Longitudinal-section view.

Fig. 3. TSV filled with Cu with a SiO_2 liner.

demonstration here, we build a two-layer 3D IC chip structure with uniform TSV distribution in COMSOL as shown in Figure 2. A popularly used TSV structure [22] with full copper filling and a silicon dioxide liner between copper and silicon applied is shown in Figure 3.

As a necessary structure in 3D IC, TSVs, however, lead to the thermal induced stress problem, which harms the reliability of the chip. There are two major reasons for the problem. For the first reason, TSV usually has a much higher thermal conductivity than silicon wafers because of the materials it used. As a result, a large temperature gradient may appear in the area close to TSV, which usually leads to large thermal stress. For the second reason, the mismatch in CTE also brings significant stress increase. The CTE of copper is $17e-6\,\mathrm{K}^{-1}$, which is nearly seven times larger

than that of silicon ($2.56e-6\,\mathrm{K}^{-1}$). When temperature increases with the same degree, the copper expansion will be much more significant than silicon, resulting in considerable stress.

The stress in solid in Cartesian coordinate can be expressed as [6]:

$$\begin{cases} -f_x = k_x - \dfrac{E\alpha}{1-2\nu} \cdot \dfrac{\partial T}{\partial x} \\ -f_y = k_y - \dfrac{E\alpha}{1-2\nu} \cdot \dfrac{\partial T}{\partial y}, \\ -f_z = k_z - \dfrac{E\alpha}{1-2\nu} \cdot \dfrac{\partial T}{\partial z} \end{cases} \tag{1}$$

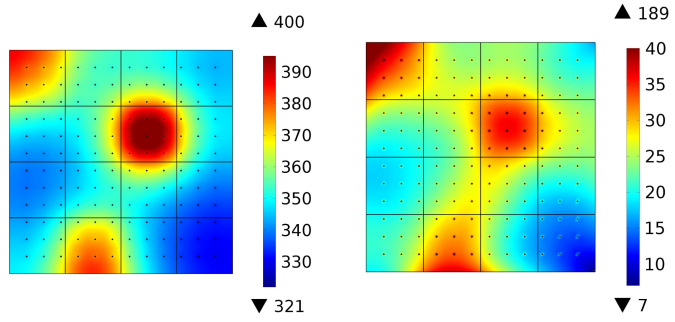
where

$$\begin{cases} k_x = \mu\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2}\right) + (\mu+\lambda)\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 v}{\partial y \partial x} + \dfrac{\partial^2 w}{\partial z \partial x}\right), \\ k_y = \mu\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2} + \dfrac{\partial^2 v}{\partial z^2}\right) + (\mu+\lambda)\left(\dfrac{\partial^2 u}{\partial x \partial y} + \dfrac{\partial^2 v}{\partial y^2} + \dfrac{\partial^2 w}{\partial z \partial y}\right), \\ k_z = \mu\left(\dfrac{\partial^2 w}{\partial x^2} + \dfrac{\partial^2 w}{\partial y^2} + \dfrac{\partial^2 w}{\partial z^2}\right) + (\mu+\lambda)\left(\dfrac{\partial^2 u}{\partial x \partial z} + \dfrac{\partial^2 v}{\partial y \partial z} + \dfrac{\partial^2 w}{\partial z^2}\right), \\ \mu = \dfrac{E}{2(1+\nu)}, \\ \lambda = \dfrac{E\nu}{(1+\nu)(1-2\nu)}. \end{cases}$$

The terms (f_x, f_y, f_z) are forces in the x, y, and z directions; (u, v, w) are displacements in the three directions; E is the elastic modulus; ν is the Poisson ratio; α is the thermal expansion coefficient; T is the temperature; and μ and λ are the Lamé coefficients. From Equation (1), it can be observed that (f_x, f_y, f_z) changes with temperature T.

As a powerful method for the analysis of thermo-mechanical stress in a complex structure where experimental investigation is quite difficult, FEM method can be used to build the 3D IC stress model based on Equation (1). We have built a two-layer 3D IC model with 12×12 TSVs uniformly placed in the whole chip using the FEM-based software COMSOL. The size of whole chip is $1\,\mathrm{cm} \times 1\,\mathrm{cm} \times 300\,\mu\mathrm{m}$, and it is divided into 4×4 same-sized blocks to represent 16 cores. Both of the two layers are $1\,\mathrm{cm} \times 1\,\mathrm{cm} \times 100\,\mu\mathrm{m}$ as shown in Figure 2. For the TSV structure, we set the values of r_i and r_o in Figure 3 as 20 μm and 24 μm, respectively. We also couple the solid heat conduction field and the solid mechanical field. The stress-free temperatures of 3D IC can be set differently according to different manufacturing processes (specifically, the common stress-free temperatures are 300 K [8], 323 K [20], and 548 K [15]). Using the FEM model, temperature and Von Mises stress information can be extracted with different power distributions as model input.

Take the bottom layer as an example. From Reference [23], it is known that the stress in the Z direction is 0, and as a result, we only need to analyze the stress in the 2D plane. Figure 4(a) and Figure 4(b) show the temperature and Von Mises stress of bottom surface for a given power distribution with stress-free temperature 300 K [8], respectively. By observation and comparison of these two figures, it is easy to find out that stress depends not only on temperature but also on TSV positions: There are many places that have *higher* temperatures but turns out to have *smaller* stresses than that of some lower temperature places, because the previous places are much further away from TSVs than the latter ones. This is an important observation that means that taking only the temperature as the optimization consideration is insufficient for reliability management methods.



(a) Temperature (K) distribution. (b) Von Mises thermal stress (MPa) distribution.

Fig. 4. Temperature and the corresponding Von Mises thermal stress distributions of the bottom surface of the 3D IC with uniform TSV distribution in COMSOL, with stress-free temperature set as 300K [8].

4 RUNTIME STRESS ESTIMATION FOR 3D IC USING ARTIFICIAL NEURAL NETWORK

In the previous section, we have shown that stress distribution in 3D IC realtes not only relates to the temperature distribution but also to the TSV distribution. As a result, it is important to take the stress information directly into account in reliability management methods. Unfortunately, the FEM-based stress analysis method is too expensive to be used for reliability management, which requires real-time stress information.

In this section, the new ANN-based runtime stress analysis method will be presented to solve the problem above. We will start with showing the basic idea that leads to the ANN-based runtime stress estimation method in Section 4.1. Then, the general framework of ANN-based stress model is presented in Section 4.2. This framework is compatible with different ANN structures, so we demonstrate three ANN stress models with different ANN structures. To be specific, the normal ANN-based stress model is presented in Section 4.3, where the potential problem of this simple ANN structure is also analyzed. An improved ANN stress model with hand-crafted feature extraction is shown in Section 4.4. Although this structure can be powerful in theory, its performance highly depends on the human experience. Finally, the CNN stress model is demonstrated in Section 4.5. With the automatic feature extraction ability specialized in image processing, the CNN stress model should be powerful in runtime stress analysis.

4.1 The Basic Idea of Building the ANN-based Stress Model

In this part, we show the basic idea of using ANN-based stress model for runtime stress estimation of 3D IC. The key observations and thinking that lead to the ultra fast ANN model-based stress analysis are presented.

First, runtime 3D IC stress analysis can be performed for individual TSV instead of the full-chip. As discussed in Section 3, TSV usually has a higher thermal conductivity than silicon wafers. As a result, a large temperature gradient may appear in the area close to TSV, resulting in large stress. In addition, large stress also tends to appear at the positions where different materials attach, that is, near TSV [2, 23]. Because of the reasons above, the largest stress always appears *not far away* from the TSV center as verified by Figure 4 and much related research [2, 15, 23]. These observations indicate there is no need to build a full-chip stress model as in Section 3. Instead, a stress model for each TSV is preferred to save computing time, especially for runtime usage where speed is preferred over accuracy.

Second, only very few important stress information like the maximum stress around each TSV is needed for runtime reliability management of IC systems. In reliability model, thermal and stress information is always chosen at the worst case, since we need such conservative condition to guarantee the absolute safety of IC systems. For example, in the reliability model considering stress migration effect, the mean time to failure (MTTF) of the IC system is calculated as [1, 32, 33]

$$MTTF_{SM} = A_0 \sigma^{-n_{sm}} e^{\frac{E_a}{k_b T_o}}, \tag{2}$$

where A_0, n_{sm}, and E_a are material-dependent constants, k_b is Boltzmann constant, and T_o is operating temperature of the IC system. We can see from Equation (2) that $MTTF_{SM}$ will be smaller with larger stress σ and larger operating temperature T_o. In other words, larger stress and temperature lead to a shorter lifetime. As a result, when the reliability model (2) is used for reliability management, σ in Equation (2) should be the maximum stress such that the MTTF is always underestimated to guarantee the absolute safety of the IC system in the management process [33]. In conclusion, runtime reliability management does not require full-chip stress information. Instead, it relies on the important stress information like the maximum stress.

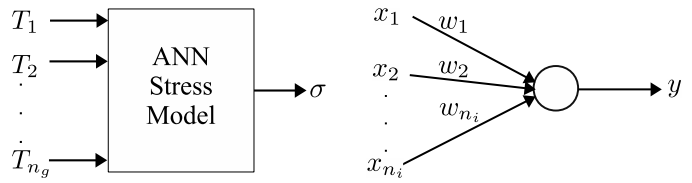
Third, ANN is a good candidate in modeling the complex relationship between the temperature input and the important stress output. ANNs are a family of statistical learning models inspired by biological neural networks (the central nervous systems of animals, in particular the brain), they are widely used to estimate or approximate functions that can depend on a large number of inputs. Especially, as a nonlinear model with complex internal connections, ANN shows significant advantages in modeling the nonlinear complex systems. Such advantages have already led to the huge success of applying ANN in computer vision, pattern recognition, speech recognition, language processing, etc. [10]. According to the previous paragraph, the input and output of the stress model are temperatures around each TSV and the important stress information of each TSV, respectively. The relationship between such input and output is not direct and highly nonlinear. As a result, as a nonlinear model specialized in modeling complex systems, ANN is chosen as the compact stress model for runtime stress estimation.

As discussed above, the maximum stress around each TSV is chosen as the output of the compact stress model to save the computing cost as it is the sufficient stress information required by 3D IC reliability management. ANN, which is good at extracting the complex connections between input and output, is used to capture the complex temperature-stress behavior in 3D IC.

4.2 The Framework of ANN-based Stress Model

In this part, the framework of the ANN-based stress model will be demonstrated, which works for all ANN structures presented in this article. The new stress model aims at calculating the maximum stress around each TSV on chip at runtime speed by taking the temperature distribution around the TSV as input.

To be specific, an ANN stress model for each TSV is built with the input–output structure as shown in Figure 5(a). The input of the model is the temperature around the TSV, written as $\{T_1, T_2, \ldots, T_{n_g}\}$, where n_g is the temperature grid number around each TSV. Because the largest stress always appears *not far away* from the TSV center as discussed in Section 4.1, taking all temperature data around one TSV as inputs introduces a lot of redundancy, which harms the stress estimation speed and accuracy. As a result, we simply ignore the temperature information for the grid elements that are further than a threshold distance, leading to significant redundancy reduction. For runtime usage, the temperature around the TSV can be obtained by runtime thermal estimation method with the help from the on-chip thermal sensors. There are many runtime thermal estimation methods proposed recently. For example, the interpolation-based method estimates the full-chip temperature with negligible computing overhead [21], and the



(a) The input-output structure of the ANN based stress model. The model is composed of neurons shown in Fig. 5b.

(b) The function of one neuron in the ANN based stress model.

Fig. 5. The framework of ANN-based stress model. All ANN stress models presented in this article share the same input–output structure and neuron function as shown in the figure.

leakage-aware thermal estimation method provides highly accurate temperature distribution at runtime [36].

The model output is the maximum stress σ. The ANN stress model is also compatible with multiple outputs, denoted as $\{\sigma_1, \sigma_2, \ldots, \sigma_{n_o}\}$, where n_o is the output number. But in this work, we focus on the single output case for simplicity, which is sufficient for reliability management at runtime.

Internally, ANNs are generally composed of interconnected "neurons," which send messages to each other. The neurons have the same structure as shown in Figure 5(b), representing the same function (assume this neuron has n_i inputs):

$$y = f\left(\sum_{i=1}^{n_i} x_i w_i + b\right), \tag{3}$$

where x_i is the ith input, w_i is the weight of x_i, b is the bias, and y is the output. f is a nonlinear function called the *activation function*. The common activation functions are tanh, sigmoid, and relu. Their formulas are expressed as follows:

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}},$$
$$f(x) = \text{sigmoid}(x) = \frac{1}{1 + e^{-x}},$$
$$f(x) = \text{relu}(x) = \max(0, x).$$

Although all neurons have the same structure, multiple neurons can compose different layers, and multiple different layers compose different complex neural networks.

The ANN stress model must be trained offline before it can be used for runtime stress estimation. To perform training, many input (temperature around the TSV) and output (maximum stress) data pairs (called samples) of the ANN stress model are first obtained (one data pair is obtained from one TSV) by measurements or detailed FEM simulations of the 3D IC with different power/temperature distributions. Then, the ANN stress model is trained using these data pairs (samples). Specifically, training means tuning the network parameters (such as weights) to minimize the difference between the ANN stress model outputs (with the sample input data as input) and the sample output data. After training, the ANN stress model can be used for runtime stress estimation. It will take the temperature data around each TSV as input, and output the maximum stress estimation for the corresponding TSV in a 3D IC. To achieve the best stress estimation accuracy, we recommend collecting the training samples from the same 3D IC on which the runtime stress estimation is performed.

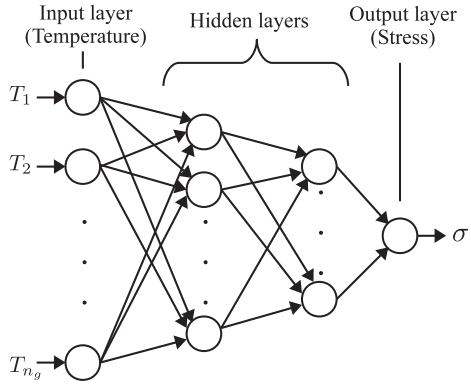


Fig. 6. A normal ANN-based stress model with two hidden layers, n_g inputs and 1 output. All the adjacent layers are fully connected in this model.

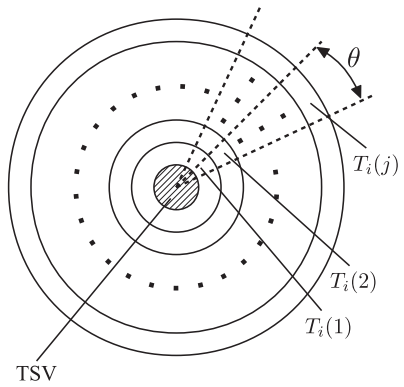


Fig. 7. The circular shaped temperature grid. Assume the plane around the TSV is divided into k pieces (each piece with an angle $\theta = 2\pi/k$ in the figure) around the full circle by the radial lines and j rings divided by the circles, making a total of $n_g = k \times j$ temperature inputs. $T_i(1), T_i(2), \ldots, T_i(j)$ represent the j temperatures of the ith piece.

With the same general framework presented above, the ANN stress model can have a variety of structures. They have different model accuracy and compactness due to their different feature extraction abilities. In the following parts, three ANN stress models with different structures will be demonstrated, including the normal ANN-based stress model (Section 4.3), ANN stress model with hand-crafted feature extraction (Section 4.4), and CNN-based stress model (Section 4.5).

4.3 The Normal ANN-based Stress Model

In general, the normal ANN-based stress model has a simple structure. It has multiple layers with different numbers of neurons in each layer. The adjacent layers in the normal ANN-based stress model are *fully connected*, i.e., the output of a neuron in the previous layer is connected to the inputs of all neurons in the next layer.

An example of this normal ANN-based stress model is given in Figure 6. The first layer is called the input layer and the last layer is called the output layer, forming the same input–output structure as the ANN stress model framework shown in Figure 5(a). The other layers are called hidden layers.

To generate the input of the stress model, we generate temperature grids in a circular shaped way as shown in Figure 7. Assume the plane around the TSV is divided into k pieces (each piece

with an angle $\theta = 2\pi/k$ in the figure) around the full circle by the radial lines, and j rings divided by the circles, making a total of $n_g = k \times j$ temperature inputs. As discussed in Section 4.2, only the temperatures of the grid elements close enough to the center are selected as inputs of the ANN.

The normal ANN stress model can be trained offline using the standard backpropagation (BP) method in conjunction with an optimization method such as gradient descent. Since this training process is standard and does not introduce runtime overhead, interested readers are referred to Reference [10] for details.

The normal ANN-based stress model has a drawback: Its input temperatures have redundancies, resulting in a less compact stress model. To be specific, the TSV and the areas around it have a *rotational symmetry property* due to its round structure: Two different thermal/stress maps can be exactly the same after a certain amount of rotation. This clearly indicates redundancy if we take these two thermal/stress maps as different samples. We will show next that such redundancy can be reduced by using the ANN stress model with hand-crafted feature extraction.

4.4 The ANN Stress Model with Hand-crafted Feature Extraction

As discussed previously, the normal ANN-based stress model presented in Section 4.3 has a problem: It has large input redundancy due to the rotational symmetry property of the TSV structure. Such large input redundancy may lead to a large-sized ANN stress model with large computing overhead or an ANN stress model with large stress estimation error.

To solve this problem, we introduce the second ANN stress model in this section. This model has the same ANN structure as the previous one, but we add a hand-crafted feature extraction procedure to the input data to reduce the redundancy in the original input.

The proposed hand-crafted feature extraction scheme is demonstrated in Figure 7. We first generate temperature grids in the same way as the normal ANN stress model presented in Section 4.3. Assuming the plane around the TSV is divided into k pieces and j rings, we can calculate the total temperature value of all sections in each piece. For example, there is $M_i = T_i(1) + T_i(2) + \cdots + T_i(j)$ for the ith piece. Then, we find the largest total temperature value and record the corresponding piece's angle, with respect to a reference line. Finally, we are able to rotate the largest total temperature piece back to the reference line for all samples and solve the rotational symmetry problem.

The remaining procedures (such as the training process) of the ANN stress model with hand-crafted feature extraction are the same as the normal ANN stress model, because they basically share the same internal network structure.

4.5 The CNN-based Stress Model

4.5.1 Motivation of Using CNN for Automatic Feature Extraction in Runtime Stress Estimation. In the previous part, we have shown an improved ANN stress model with hand-crafted feature extraction scheme. However, building such a model requires human observation and experience in finding the redundancies inside the input to extract the key features. Since there can be a variety of redundancies and features, it is extremely difficult to design an efficient feature extraction scheme based on human experience. Taking the hand-crafted feature extraction scheme in Section 4.4 as an example, it still has two drawbacks.

First, the temperature data are reshaped and flattened as a vector, which is then fed to the ANN stress model as the input. However, the temperature information around each TSV clearly has spatial correlations in the two-dimensional space on the horizontal plane. Within the plane, there are correlation and gradient related features in the temperature data, but flattening these temperature data into a vector makes extracting these spatial features even more difficult.

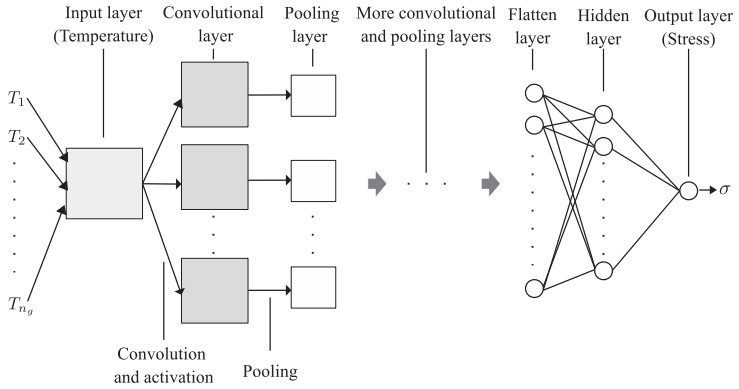


Fig. 8. Structure of the CNN-based stress model. Each square in the figure represents a data matrix, which is also called feature map in convolutional layer and pooling layer. The convolutional layer includes both convolution and activation operations. In each convolutional/pooling layer, there are multiple convolution/pooling operations working in parallel. There can be multiple convolutional and pooling layers appearing alternately in the CNN stress model shown as "More convolutional and pooling layers" in the figure. The input and output of the CNN stress model follow the ANN stress model framework shown in Figure 5(a), except that the input of the CNN stress model is stored in matrix form rather than in vector form.

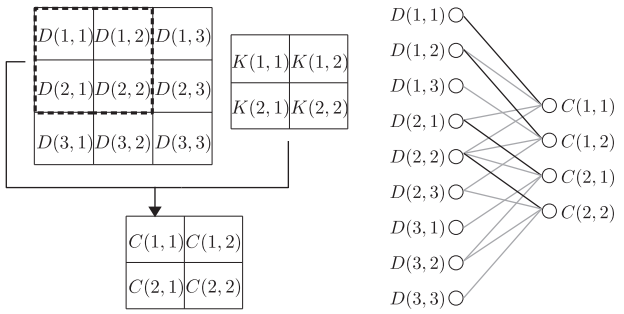
Second, the hand-crafted feature extraction procedure could be much more complex than exploiting the rotational symmetry property. Although we know considering the rotational symmetry only is insufficient to unleash the full potential of the ANN stress model, designing a more advanced feature extraction scheme by observation is too difficult for human.

Based on the reasons above, we further analyze the performance of the ANN stress model with automatic feature extraction. Specifically, we introduce the CNN [18] to perform the automatic feature extraction at the input stage of the ANN stress model. With the CNN structure, the two-dimensional features can be extracted, leading to a more accurate stress estimation.

CNN is specialized in handling data with a gridlike pattern, because it can process data in the plane by extracting the distribution and gradient related features automatically [10]. Because of this, CNN has been tremendously successful in applications such as image processing and pattern recognition.

Similarly to image processing, stress estimation also deals with gridlike data input, since the temperature grids around each TSV are just like a picture. We can even convert the temperature data into a picture by replacing the temperature values with standard gray-scale values. As a result, CNN model, which specializes in extracting features from two-dimensional gridlike data, can be applied to our fast stress estimation problem at the input feature extraction stage.

4.5.2 The General Structure of the CNN-based Stress Model. CNN is defined as the neural network that implements convolution operation in at least one of its layers. The structure of CNN used for runtime stress estimation in this work is shown in Figure 8, with the following data processing flow. First, we import the same input temperature data as in the ANN stress model framework into the CNN stress model. Second, the convolutional layer and pooling layer will process the input temperature data and extract its key features as the output data matrices called *feature maps*. Note that there are multiple convolution/pooling operations working in parallel in each convolutional/pooling layer. We can also repeat these two steps by adding multiple convolutional and pooling layers. Third, a flatten layer will flatten the feature maps in matrix form into vector form, which is then fed into a fully connected neural network. Finally, the maximum stress around the TSV is obtained as the output of the fully connected neural network.



(a) The convolution operation in matrix form. The kernel size is 2×2 in this example.

(b) The convolution operation demonstrated using neurons. All the black lines represent connections with shared weight $K(1,1)$.

Fig. 9. Illustration of the convolution process in the convolutional layer of the CNN stress model.

Next, we will look into each layer of the CNN-based stress model and specially focus on the two important layers: the convolutional layer and the pooling layer. The functional details of each layer in the stress estimation process will be illustrated.

4.5.3 The Input Layer of the CNN Stress Model. First, at the input layer shown in Figure 8, temperature distribution data at n_g positions around the TSV is directly input as a $n_d \times n_d$ temperature matrix (with $n_g = n_d^2$). This is because the CNN operates directly on the two-dimensional matrix data (or high-dimensional tensors in general), unlike the normal ANN, which operates on vector data. The element value in the input temperature matrix is the temperature value at the corresponding spatial position around the TSV, which is similar to the gray-scale value of a picture.

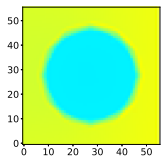
4.5.4 The Convolutional Layer and Its Properties. The convolutional layer in Figure 8 performs convolution operation, which is one of the key operations in the CNN stress model. The main purpose of performing convolution in stress estimation is to eliminate the redundancies in the input matrix and storing only the important information into a smaller output matrix called feature map. The convolution operation can be expressed as the following equation:

$$C(i,j) = \sum_{p=1}^{n_k} \sum_{q=1}^{n_k} K(p,q) \cdot D(i+p-1, j+q-1), \tag{4}$$

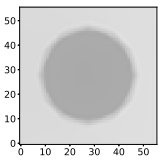
where C is the output feature map matrix with $C(i,j)$ as its element at the ith row and jth column (we also use the same representation in other matrices like K). $K \in \mathbb{R}^{n_k \times n_k}$ is the convolution kernel matrix, with *kernel size* $n_k \times n_k$. D is the input matrix of convolution operation.[1] To extract different features of the input data, usually there are multiple convolution operations with different convolution kernels working in parallel on the input data, as shown in Figure 8.

The convolution process is demonstrated in Figure 9. From this figure, it is clear that the convolution process, usually represented in matrix form (as in Figure 8), is also composed of the basic neuron structure shown in Figure 5(b). There are two special properties of the convolutional layer compared to the normal ANN layer. First, the convolutional layer has sparse connectivity, while

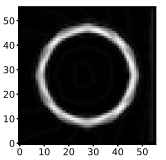
[1]For the first convolutional layer of the CNN stress model, $D \in \mathbb{R}^{n_d \times n_d}$ is composed of temperatures around the TSV, i.e., $T_1, T_2, \ldots, T_{n_g}$, where $n_g = n_d^2$.



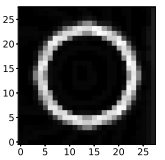
(a) Temperature distribution around a TSV.



(b) The grey-scale like data map of the temperature distribution.



(c) The feature map after the convolution operation.



(d) The feature map after pooling. With pooling window size 2×2, the data element number is reduced to 1/4 of the feature map shown in Fig. 10c by pooling.

Fig. 10. A data map processing example in convolutional layer and pooling layer.

normal ANN is fully connected. For example, $D(1,1)$ in the convolutional layer is only connected to $C(1,1)$. Second, there are weight shares for different connections in the convolutional layer, while the weights of different connections are generally different for normal ANN. This also means convolutional layer usually has much fewer weights than a normal ANN layer. To be specific, as defined in Equatoin (4), the weights in convolution are actually the convolution kernel matrix elements. For the example shown in Figure 9, there are only four weights ($K(1,1)$, $K(1,2)$, $K(2,1)$, and $K(2,2)$) in the convolutional layer, and weight $K(1,1)$ is shared by four connections ($D(1,1)$ to $C(1,1)$, $D(1,2)$ to $C(1,2)$, $D(2,1)$ to $C(2,1)$, and $D(2,2)$ to $C(2,2)$).

The two properties presented above lead to several advantages of the CNN stress model in fast stress estimation. First, stress estimation using CNN is faster than using a normal ANN structure, due to the sparse connectivity property in CNN. Equivalently, it also means we can build a larger and deeper network using CNN than using normal ANN, with the same runtime overhead. In addition, the weight sharing property means the storage requirement is smaller for CNN with the same size.

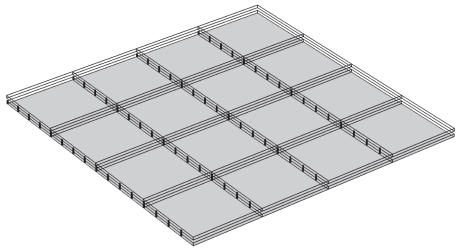
More importantly, the two properties lead to the fact that convolution is very efficient in extracting the features of the input temperature data, especially in detecting the shape of the large spatial gradients (which is also called detecting the *edge features*) [10]. Since stress is caused by large temperature gradient as discussed in Section 3, large stress tends to appear in places with large temperature gradients, or in other words, with obvious edge features. These edge features can be extracted by convolution efficiently.

Now, we employ a specific example in Figure 10 to show the convolution process in edge feature extraction. The temperature data map around a TSV is presented in Figure 10(a), which is equivalent to a gray-scale image shown in Figure 10(b). Subsequently, the convolution operation is performed on the original temperature data map with a trained convolution kernel. The feature map after convolution is given in Figure 10(c), with edge features clearly visible.

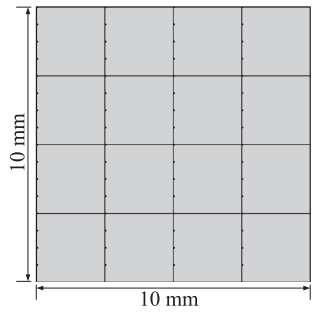
After convolution, activation is applied to the output matrix in the convolutional layer. This is a standard operation to enable the nonlinear modeling ability of neural network as presented before in Section 4.2.

4.5.5 The Pooling Layer and Its Properties. Now we present the pooling layer, which always appears after the convolutional layer as shown in Figure 8.

The pooling layer further shrinks the size of its input feature map (which is also the output feature map of the convolutional layer). This process is done by outputting only one single statistic for several adjacent data elements in the input feature map. Specifically, each output element of

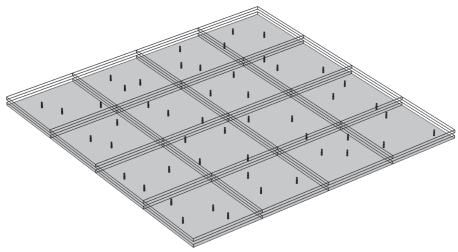


(a) Side view showing the two die layers connected with 64 TSVs.

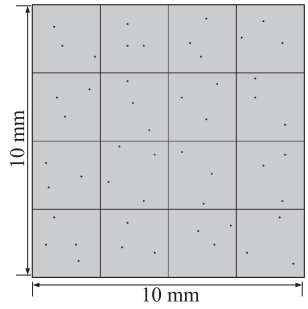


(b) Bottom view of chip showing the 16-core architecture for each layer.

Fig. 11. A 32-core (16 cores on each layer) 3D IC with nonuniform TSV distribution built in COMSOL. Package structure is not shown in the figure.



(a) Side view showing the two die layers connected with 50 TSVs.



(b) Bottom view of chip showing the 16-core architecture for each layer.

Fig. 12. A 32-core (16 cores on each layer) 3D IC with random TSV distribution built in COMSOL. Package structure is not shown in the figure.

the pooling layer is computed from $n_p \times n_p$ adjacent elements of a local square within the input feature map, by using a function such as taking the average (called the *average pooling*) or taking the maximum (called the *max pooling*) of these adjacent input elements, where $n_p \times n_p$ is called the *pooling window size*. The input elements for different output elements do not overlap. So the output feature map of the pooling layer is n_p^2 times smaller than its input feature map.

The main purpose of using pooling operation in stress estimation is to make the features invariant to small position changes in the input feature map, such that the exact position information is dropped as redundant information. In stress estimation, we usually care much more about the presence of the features than the *exact* positions of these features. For example, the maximum stress around the TSV, which is the final output of the stress model, is determined by a local temperature pattern feature rather than such feature's exact location. Thus, the exact feature position information can be dropped by using the pooling operation without effecting the final output accuracy, leading to a much smaller output feature map after pooling. More importantly, the max pooling operation makes the CNN structure immune to the rotational symmetry problem in the

Table 1. Sample Numbers for Training and Validation of Each Stress-free Temperature

Sample type	TSV	Power	Sample #	Total size
Training	Uniform	Synthetic	11,520	225 MB
Validation	Uniform	Synthetic	2,880	56 MB
	Uniform	SPEC	2,880	56 MB
	Nonuniform	Synthetic	3,200	63 MB
	Nonuniform	SPEC	3,200	63 MB
	Random	Synthetic	3,000	61 MB
	Random	SPEC	3,000	61 MB

Table 2. General Hyperparameters for Training and Validation of the ANN Stress Models

Parameter name	Settings
Batch size	10
Learning rate	0.1
Training epoch	5

Table 3. Stress Estimation Accuracy Results of the Normal ANN-based Stress Model on 3D IC with Uniform TSV Distribution in Figure 2

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	7.10	9.88	9.71	35.6	37.6	39.8	7.16	9.85	10.28	37.8	35.9	40.3
40	6.48	9.46	9.78	24.8	34.3	34.6	6.51	9.43	9.81	33.5	29.7	36.2
60	5.94	7.35	7.65	27.7	32.8	28.7	5.97	7.53	7.74	25.7	31.4	30.1
140	5.96	7.46	8.31	20.5	27.1	36.5	6.14	7.51	8.33	26.1	27.6	32.4
200	6.07	8.62	9.05	22.3	34.5	32.4	6.08	8.57	8.95	21.5	33.2	29.8
40; 10	5.50	8.89	7.71	25.7	33.6	33.6	5.73	8.89	7.63	19.8	27.1	25.5
55; 15	4.94	5.67	6.38	17.2	20.7	24.7	5.11	5.44	6.40	22.3	24.0	21.9
70; 20	5.41	6.98	7.56	25.4	23.4	32.8	5.75	7.02	7.62	21.1	28.3	24.6
90; 40	6.00	7.14	8.70	21.6	26.9	24.8	6.11	7.54	8.77	19.3	26.5	31.7
140; 60	5.99	7.51	7.25	19.7	21.7	26.7	6.03	7.33	7.42	23.8	30.2	28.6
200; 80	6.31	8.95	7.07	21.4	32.4	31.5	6.52	9.11	7.21	23.2	34.8	27.9
40; 10; 5	6.32	8.37	7.89	21.9	29.4	32.6	6.35	8.24	8.01	24.9	27.8	29.1
55; 15; 10	6.05	5.97	6.86	24.8	25.1	30.5	6.11	6.02	7.01	21.1	23.9	25.0
80; 30; 15	5.68	6.88	6.73	18.3	32.6	34.1	5.62	7.03	6.45	19.7	27.1	22.8
120; 60; 20	5.36	7.76	7.39	23.5	31.2	33.8	5.63	8.00	7.14	26.1	29.3	30.5
200; 80; 30	5.26	9.62	9.71	19.1	36.3	34.6	5.66	8.83	8.59	31.5	28.4	29.1
60; 30; 15; 5	15.4	18.2	18.9	45.6	43.9	50.1	13.9	16.7	17.5	55.3	49.8	57.2
200; 120; 60; 20	18.7	17.3	16.7	52.1	51.2	52.3	17.9	18.5	15.5	49.2	54.7	58.3

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

Table 4. Stress Estimation Accuracy Results of the Normal ANN-based Stress Model on 3D IC with Nonuniform TSV Distribution in Figure 11

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	7.53	10.1	10.0	41.2	40.3	42.8	7.36	9.99	10.0	38.4	32.5	36.1
40	7.01	9.65	10.6	30.5	31.8	38.9	6.78	9.63	9.97	34.6	30.8	32.5
60	6.81	8.13	7.74	35.6	29.9	30.5	6.13	7.69	7.84	26.8	33.5	28.1
140	6.32	7.82	8.75	23.7	26.3	31.8	6.36	7.76	8.54	25.8	29.0	31.6
200	6.53	8.33	8.87	27.9	33.5	29.0	6.29	8.53	8.87	23.9	30.4	27.3
40; 10	5.87	9.01	7.98	28.2	31.4	27.1	5.92	8.76	7.89	19.8	27.1	25.6
55; 15	5.27	5.83	6.55	19.4	23.9	21.8	5.26	5.39	6.48	19.0	25.7	23.4
70; 20	5.83	7.34	7.74	29.7	38.6	26.4	5.86	7.32	7.74	23.2	24.7	28.9
90; 40	6.35	7.48	8.97	31.5	27.9	34.0	6.41	7.60	8.59	23.8	28.6	30.2
140; 60	6.38	7.88	7.73	33.2	28.5	24.1	6.51	7.48	7.36	25.6	29.1	27.3
200; 80	6.84	9.65	7.56	25.7	29.4	36.6	6.83	9.01	7.35	27.1	32.3	27.6
40; 10; 5	6.69	8.55	8.11	23.2	27.5	31.2	6.68	8.41	8.15	26.1	28.7	30.8
55; 15; 10	6.58	6.34	7.10	20.8	32.4	28.5	6.52	6.16	7.33	23.7	25.9	27.3
80; 30; 15	5.97	7.16	6.99	28.9	24.7	31.4	5.93	7.18	6.52	20.9	25.8	25.7
120; 60; 20	5.78	8.15	7.40	31.4	28.4	26.8	5.85	8.41	7.37	25.5	30.4	32.6
200; 80; 30	5.80	8.92	7.20	35.1	38.5	28.4	5.90	8.87	8.63	37.6	31.5	29.7
60; 30; 15; 5	15.5	18.7	17.1	55.9	57.8	47.2	16.4	17.7	16.9	52.4	58.1	53.9
200; 120; 60; 20	18.9	17.6	18.3	56.0	46.3	60.8	18.1	18.6	17.3	56.7	55.0	50.8

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

TSV structure presented previously in Section 4.3, because it has been proved that max pooling over spatial positions is naturally invariant to translation [11].

Now, we continue to use the example in Figure 10 to illustrate the effect of the pooling operation on stress estimation. We process the input feature map shown in Figure 10(c) using max pooling operation (with pooling window size 2×2), and then the output feature map after pooling is shown in Figure 10(d). The two feature maps (before and after pooling) look very similar to each other, meaning the pooling operation successfully preserved the important features of the input feature map. Furthermore, the output feature map in Figure 10(d) is four times smaller (with 28×28 data elements) than the input feature map in Figure 10(c) (with 56×56 data elements). This is because the pooling operation reduces the redundancies in the input feature map by dropping the exact feature position information. This example proves the benefits of pooling: The feature map size is reduced while the key features in the original feature map are well preserved.

4.5.6 The Rest of the Layers after the Pooling Layer. As shown in Figure 8, after pooling, the flatten layer will transform the two-dimensional feature maps into a data vector. The data vector will be further processed by a fully connected ANN. Finally, the maximum stress will be outputted by the fully connected ANN, which completes the whole stress estimation process using the CNN-based stress model.

5 EXPERIMENTAL RESULTS

In this section, we present the experimental results to show the performance of the fast stress estimation method for 3D IC using ANN-based stress models.

Table 5. Stress Estimation Accuracy Results of the Normal ANN-based Stress Model on 3D IC with Random TSV Distribution in Figure 12

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	7.48	10.3	9.8	36.7	42.5	43.1	7.45	9.83	10.7	41.2	37.5	33.1
40	7.15	9.33	10.2	32.4	34.8	36.9	6.55	9.72	9.54	33.1	37.8	30.2
60	6.68	8.41	7.59	38.2	27.9	33.7	6.08	7.71	7.56	23.9	30.5	28.8
140	6.48	7.73	8.66	25.9	22.4	30.1	6.42	7.58	8.33	22.7	28.3	30.3
200	6.24	8.10	8.57	23.5	30.4	28.2	6.29	8.46	8.62	24.7	31.5	26.8
40; 10	5.51	9.21	7.83	26.7	30.4	28.3	5.87	8.65	7.93	20.6	28.7	27.3
55; 15	5.24	5.91	6.49	18.9	24.6	22.3	5.31	5.34	6.46	18.7	23.8	22.3
70; 20	5.91	7.47	7.68	28.2	33.5	27.1	6.11	7.23	7.65	21.3	25.1	27.6
90; 40	6.38	7.39	8.86	32.4	29.6	32.1	6.33	7.46	8.36	22.4	27.9	25.3
140; 60	6.33	7.92	7.78	33.6	24.7	25.6	6.38	7.79	7.46	27.3	28.4	29.2
200; 80	6.59	9.31	7.68	23.9	28.6	32.1	6.97	9.15	7.21	25.1	30.8	26.8
40; 10; 5	6.58	8.64	8.23	24.9	25.8	30.3	6.63	8.39	8.37	27.3	26.5	31.6
55; 15; 10	6.46	6.55	7.13	20.6	32.3	27.1	6.34	6.05	7.28	24.2	24.3	26.4
80; 30; 15	5.86	7.33	6.82	28.6	23.1	28.3	5.97	7.22	6.72	21.8	27.3	28.1
120; 60; 20	5.69	8.23	7.38	32.0	27.6	27.3	5.66	8.72	7.43	24.6	28.3	30.1
200; 80; 30	5.72	8.83	7.16	34.3	36.7	29.8	5.83	8.66	8.72	38.1	32.0	28.9
60; 30; 15; 5	14.6	17.3	18.2	56.7	58.6	52.3	14.6	16.1	18.3	54.8	57.3	59.7
200; 120; 60; 20	19.3	18.4	17.6	58.1	49.6	57.3	16.1	17.8	15.9	54.9	58.2	53.7

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

5.1 Experimental Settings

The training and validation procedures of all ANN stress models are performed using the open source deep learning toolbox Apache MXNet framework [5] on a laptop PC with 2.50GHz dual-core four-thread CPU and 8GB memory.

The stress-free temperature of 3D IC can be different depending on different manufacturing processes. In this experiment, we set three common stress-free temperatures as 300K [8], 323K [20], and 548K [15].

The temperature and stress data for training and validation in this experiment are obtained from 3D IC models built in COMSOL 5.0. To be specific, the training data are generated from 3D IC with 144 uniformly distributed TSVs (see Figure 2) as presented previously in Section 3. The validation data are obtained from three 3D ICs: the first one with the same structure as the one used to generate training data (see Figure 2), the second 3D IC model with a common nonuniformly TSV distribution containing 64 TSVs [24] (see Figure 11), and the third 3D IC model with random TSV distribution containing 50 TSVs (see Figure 12). All the 3D ICs share the same TSV structure as shown in Section 3.

The training and validation sample numbers are summarized in Table 1, where each sample contains the input temperature distribution data and the corresponding stress distribution data around a TSV. As shown in the table, for each stress-free temperature, we get 11,520 training data samples (with a total data size of 225MB) by simulating the 3D IC model in COMSOL with different power distributions. Specifically, the power distributions for training are created manually (called synthetic power), which should be diverse enough to cover the power range from

Table 6. Stress Estimation Accuracy Results of the ANN Model with Hand-crafted Feature Extraction on 3D IC with Uniform TSV Distribution in Figure 2

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	5.53	7.56	7.60	25.4	21.9	32.7	5.61	7.49	7.78	28.1	30.3	31.9
40	5.57	7.21	6.41	21.0	29.7	34.1	5.63	7.31	6.38	23.5	26.7	25.4
60	4.97	4.86	5.74	17.4	22.6	25.3	4.83	5.21	5.73	21.5	27.0	31.2
140	4.23	6.23	3.99	22.3	25.2	18.2	4.21	6.15	4.17	19.2	25.4	21.8
200	4.12	6.64	4.68	23.1	28.1	25.2	4.24	6.68	4.59	16.7	21.3	23.0
40; 10	3.66	4.97	5.92	15.9	23.5	23.8	3.72	4.85	5.98	14.8	19.6	21.9
55; 15	3.24	4.26	3.17	16.0	16.9	15.8	3.32	4.41	3.26	13.2	16.7	12.5
70; 20	3.53	4.69	3.77	21.8	18.5	18.3	3.58	4.71	3.68	15.6	18.2	14.7
90; 40	3.86	5.18	5.41	23.6	16.7	25.4	3.74	5.09	5.52	12.9	20.1	19.6
140; 60	4.27	6.62	6.29	18.7	21.8	29.8	4.36	6.43	6.34	18.4	25.2	23.4
200; 80	4.13	5.83	6.68	17.9	23.9	24.2	4.78	5.92	6.47	19.6	23.8	30.8
40; 10; 5	4.40	4.51	4.27	20.2	19.8	19.1	4.43	4.48	5.61	16.7	18.3	22.6
55; 15; 10	4.60	4.46	5.49	19.3	25.7	19.8	4.67	4.73	5.52	18.6	21.9	23.2
80; 30; 15	4.09	5.23	3.61	25.8	21.4	23.5	4.53	5.38	3.89	17.3	26.5	17.1
120; 60; 20	4.53	6.27	3.51	19.3	23.6	20.1	4.79	6.01	3.97	21.5	24.0	17.3
200; 80; 30	4.21	6.50	5.98	23.7	22.9	25.3	4.71	6.48	5.82	22.9	30.2	27.5
60; 30; 15; 5	14.6	18.5	17.4	40.5	42.3	49.6	13.5	17.9	19.1	40.9	52.1	49.3
200; 120; 60; 20	17.7	17.2	18.7	42.9	51.0	45.3	17.3	18.2	16.5	43.7	55.6	52.8

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

the low power (idle state of core) to the high power (high-performance state of the core) for each core. In this work, we randomly set the power of each core to be between 0 W and 3 W, which leads to a wide temperature range of 293K to 393K (20 °C to 120 °C) to cover the temperature range of everyday use of the chip. We also generate 18,160 data samples (with a total data size of 360MB) for validation. To test the generality of the new ANN stress models, the validation data samples are generated in different ways. First, we generate 2,880 samples using the 3D IC with uniform TSV distribution and synthetic power ("Uniform" TSV distribution and "Synthetic" power in Table 1). Second, we generate 2,880 samples using the 3D IC with uniform TSV distribution and SPEC CPU 2000/2006 benchmark power [12, 13], where the benchmark powers are randomly assigned to cores ("Uniform" TSV distribution and "SPEC" power in Table 1). In a similar way, we obtain 3,200 samples using the 3D IC with nonuniform TSV distribution and synthetic power ("Nonuniform" TSV distribution and "Synthetic" power in Table 1), 3,200 samples using the 3D IC with nonuniform TSV distribution and SPEC CPU 2000/2006 benchmark power ("Nonuniform" TSV distribution and "SPEC" power in Table 1), 3,000 samples using the 3D IC with random TSV distribution and synthetic power ("Random" TSV distribution and "Synthetic" power in Table 1), and 3,000 samples using the 3D IC with random TSV distribution and SPEC CPU 2000/2006 benchmark power ("Random" TSV distribution and "SPEC" power in Table 1).

For each sample, we only take the COMSOL temperature data within 0.14mm to the center of this TSV as explained in Section 4.1. Because COMSOL has different data grid from the input data grids required by our ANN stress models, we use interpolation to solve this problem. All ANN stress models have the same input data size $n_g = 784$. Specifically, the input data grid is formed

Table 7. Stress Estimation Accuracy Results of the ANN Model with Hand-crafted Feature Extraction on 3D IC with Nonuniform TSV Distribution in Figure 11

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	5.99	8.21	7.96	31.8	28.6	33.5	5.82	7.81	7.94	27.3	31.5	32.6
40	5.87	7.84	6.87	27.1	33.5	27.9	5.87	7.62	6.49	25.8	23.2	28.7
60	5.81	6.56	5.83	29.3	31.4	27.4	4.98	5.59	6.03	24.4	26.1	27.8
140	4.68	6.89	4.57	23.5	35.7	25.1	4.55	6.34	4.29	18.6	27.3	20.4
200	4.53	6.78	5.11	27.9	33.3	27.1	4.59	6.86	4.73	20.1	22.7	25.3
40; 10	4.11	5.58	6.15	16.4	31.2	29.8	3.91	5.07	6.31	15.9	17.4	25.6
55; 15	3.52	4.47	3.62	18.1	19.4	16.9	3.48	4.49	3.47	12.9	18.3	15.9
70; 20	4.14	4.89	4.23	23.6	25.6	27.0	3.83	4.94	3.79	14.8	17.9	16.1
90; 40	3.98	5.47	5.66	23.1	31.0	26.8	3.96	5.41	5.78	17.2	23.5	27.3
140; 60	4.75	6.98	6.55	19.8	33.2	27.4	4.59	6.72	6.61	17.3	22.9	26.5
200; 80	4.54	6.11	6.68	23.0	25.5	31.2	5.21	5.87	6.68	23.5	24.7	28.8
40; 10; 5	4.82	4.60	4.72	21.7	18.6	24.2	4.69	4.38	5.51	15.7	18.7	22.0
55; 15; 10	4.99	4.66	5.80	25.8	27.1	28.0	4.81	5.12	5.73	18.8	23.8	21.5
80; 30; 15	4.78	6.16	7.00	19.3	31.8	27.9	4.97	5.46	4.32	20.1	20.9	19.1
120; 60; 20	5.78	6.81	4.53	23.8	30.1	24.8	5.13	6.23	4.61	23.5	24.6	18.3
200; 80; 30	4.56	6.83	6.32	21.6	28.9	25.1	4.94	6.76	5.94	24.6	30.1	24.4
60; 30; 15; 5	15.9	19.4	18.1	53.7	60.3	54.6	14.1	15.2	17.1	45.9	55.6	53.6
200; 120; 60; 20	16.2	18.6	17.9	49.2	56.3	58.1	18.2	17.1	16.9	51.4	49.8	57.9

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

by 16 pieces and 49 rings for the normal ANN stress model and the ANN stress model with hand-crafted feature extraction, while the input data grid is 28×28 for the CNN stress model.

The activation function for all ANN stress models is chosen as tanh, which shows accuracy advantage over the other two popular activation functions sigmoid and relu. The optimization algorithm used for training is gradient descent, which has good convergence speed.

The general hyperparameters used to train and verify the ANN stress models in this experiment are collected in Table 2. In this table, batch size, learning rate, and training epoch are parameters for gradient descent, which are determined by trial and error.

5.2 Accuracy and Computing Overhead Tests of the ANN-based Stress Models

In this part, we will test the three ANN-based stress models and compare their performances in both stress estimation accuracy and computing overhead.

5.2.1 Performance of the Normal ANN-based Stress Model. We first test the performance of the normal ANN-based stress model, whose structure is presented in Section 4.3. To see the performance of the model with different sizes, we build several models with different hidden layer numbers and different neuron numbers in each hidden layer. Please note all models must have the same number of neurons in the input layer and the output layer.

The stress estimation accuracy results of the normal ANN-based stress models with different sizes are listed in Table 3, with validation samples generated from the 3D IC with uniform TSV distribution. The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column. It is noted that the best accuracy result is achieved by the

Table 8. Stress Estimation Accuracy Results of the ANN Model with Hand-crafted Feature Extraction on 3D IC with Random TSV Distribution in Figure 12

Hidden layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
20	5.87	8.34	7.65	30.3	31.5	29.7	5.77	7.79	7.98	31.4	29.1	34.7
40	5.96	7.61	6.93	23.5	31.8	28.7	5.67	7.83	6.55	31.0	26.2	25.9
60	5.77	6.43	5.72	28.1	27.6	29.7	4.88	5.68	6.23	27.5	22.1	29.3
140	4.74	6.73	4.66	25.8	31.2	27.1	4.63	6.24	4.76	21.8	24.3	23.5
200	4.47	6.83	5.32	26.5	30.3	28.7	4.67	6.58	4.29	24.1	23.5	26.8
40; 10	4.23	5.75	6.31	18.4	28.9	27.8	3.96	5.11	6.28	18.3	19.1	23.6
55; 15	3.49	4.56	3.58	19.3	20.6	17.9	3.50	4.46	3.38	14.1	16.5	16.1
70; 20	4.36	4.71	4.48	24.5	20.6	23.8	3.88	4.85	3.76	16.2	18.4	17.9
90; 40	3.97	5.61	5.58	22.5	26.7	23.9	3.89	5.31	5.65	18.3	24.5	26.2
140; 60	4.64	6.79	6.63	16.5	27.1	26.4	4.78	6.34	6.18	18.5	24.1	27.6
200; 80	4.84	6.02	6.57	21.8	24.3	27.6	5.41	5.68	6.42	24.9	23.4	27.6
40; 10; 5	4.75	4.93	4.91	21.0	19.3	27.1	4.78	4.58	5.67	16.2	17.5	21.9
55; 15; 10	4.86	4.73	5.52	24.2	26.8	25.6	4.91	5.08	5.63	21.3	23.4	22.7
80; 30; 15	4.69	6.08	7.12	20.6	27.5	27.3	4.84	5.53	4.49	21.6	23.9	25.3
120; 60; 20	5.85	6.73	4.79	24.6	29.4	27.3	5.40	6.48	4.72	22.8	26.5	19.8
200; 80; 30	4.47	6.83	6.47	21.3	27.2	26.5	4.88	6.65	5.74	23.1	31.2	22.9
60; 30; 15; 5	16.5	16.9	19.3	55.8	61.9	57.3	13.9	16.8	15.3	52.3	49.2	57.6
200; 120; 60; 20	17.2	17.5	14.8	47.3	52.1	56.9	17.6	15.4	17.2	53.9	49.7	56.2

The neuron numbers in different hidden layers are separated by semicolon in the "Hidden layer configuration" column; 300K, 323K, and 548K denote the stress-free temperatures.

Table 9. The Shared Configuration Settings for All CNN Stress Models

Parameter name	Settings
Pooling type	Max pooling
Pooling window size	2×2
Neuron # in hidden layer	15

ANN with two hidden layers configured as "55; 15", with average error to be within 7% for all test cases. Further increasing the neuron numbers in the hidden layers or increasing hidden layer number does not necessarily lead to more accurate stress estimation, because of overfitting.

From the table, we also observe that the normal ANN-based stress model shows similar accuracy for synthetic power and SPEC benchmark power, even with the fact that SPEC benchmark power data are not used in the training process. This indicates the trained network using synthetic power data is general enough to work with the SPEC benchmark power.

We also test the normal ANN-based stress model using validation samples generated from 3D ICs with nonuniform and random TSV distributions. The corresponding accuracy test results are shown in Table 4 and Table 5. By comparing the results in Table 3, Table 4, and Table 5, we see that the errors of testing on 3D ICs with both nonuniform and random TSV distributions are slightly larger than those on 3D IC with uniform TSV distribution. This is because the normal ANN-based stress model is trained using only the samples generated from the 3D IC with uniform

Table 10. Stress Estimation Accuracy Results of the CNN-based Stress Models on 3D IC with Uniform TSV Distribution in Figure 2

Convolutional layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
5 × 5, 3	3.08	4.35	3.97	17.2	17.6	15.8	3.24	4.39	4.21	21.5	16.3	19.8
5 × 5, 6	2.86	3.13	3.22	18.2	14.8	11.5	2.99	3.32	3.19	15.6	13.4	12.0
5 × 5, 10	2.31	2.83	2.76	11.6	13.1	9.59	2.39	2.91	3.02	11.9	17.5	14.1
5 × 5, 15	2.09	3.55	3.39	9.52	12.9	14.7	2.19	3.00	2.78	13.5	11.9	10.2
5 × 5, 3; 5 × 5, 6	1.94	2.93	2.76	9.43	10.5	11.9	2.08	3.12	2.78	13.5	11.9	9.31
5 × 5, 6; 5 × 5, 15	1.92	2.41	2.19	9.68	8.92	9.33	2.00	2.39	2.38	12.7	14.2	9.86
5 × 5, 10; 5 × 5, 25	1.85	1.58	1.64	7.38	7.97	8.53	1.91	1.74	1.83	9.31	10.8	8.72
5 × 5, 15; 5 × 5, 30	2.46	3.26	2.26	10.7	13.8	11.1	2.37	3.35	2.26	14.0	11.9	9.31
3 × 3, 3	2.38	3.74	3.38	11.3	14.6	11.9	2.49	3.43	3.56	11.2	14.9	13.6
3 × 3, 6	2.46	3.22	2.80	15.0	14.1	12.8	2.55	3.56	3.12	10.8	12.5	9.97
3 × 3, 10	2.69	2.79	2.56	13.7	12.4	10.9	2.71	3.02	2.50	11.5	10.8	14.7
3 × 3, 15	1.96	3.82	2.75	11.4	11.9	13.2	2.38	4.00	2.88	9.73	14.6	13.1
3 × 3, 3; 3 × 3, 6	2.13	3.97	3.63	10.8	12.0	11.3	2.32	3.68	3.74	8.92	12.5	11.0
3 × 3, 10; 3 × 3, 25	2.28	2.50	2.41	9.87	10.2	10.9	2.20	2.57	2.55	14.2	13.7	11.4
3 × 3, 15; 3 × 3, 30	1.96	3.39	2.24	13.2	10.8	8.97	2.32	3.71	2.38	11.3	12.1	14.6
3 × 3, 3; 3 × 3, 6; 3 × 3, 10	2.48	3.76	2.86	13.1	14.5	10.1	2.53	3.72	3.01	12.5	16.1	12.4
3 × 3, 6; 3 × 3, 15; 3 × 3, 20	2.62	3.34	2.53	15.4	12.9	9.38	2.73	3.46	2.59	13.6	18.4	14.7

The shared configurations of all CNN stress models are given in Table 9. The configuration parameters in different convolutional layers are separated by semicolon in the "Convolutional layer configuration" column, with "$i \times i$, j" meaning the kernel size is $i \times i$ and the convolution number working in parallel is j in the corresponding layer; 300K, 323K, and 548K denote the stress-free temperatures.

TSV distribution. Even so, the accuracy of the normal ANN-based stress model is still good for the 3D ICs with nonuniform and random TSV distributions: The ANN with two hidden layers configured as "55; 15" still achieves an average error that is smaller than 7% for all test cases. This indicates the normal ANN-based stress model even works on other 3D ICs with different TSV distributions, with slightly larger errors. Nevertheless, we still recommend to train the ANN stress model using samples from the exact 3D IC structure to achieve the best accuracy.

The runtime results of the normal ANN-based stress models are collected later in Table 13 together with the other two ANN-based stress models. The computing overhead increases as the ANN model size increases. For the ANN model with the best accuracy (with two hidden layers configured as "55; 15"), the stress estimation only takes 0.17ms, which is extremely fast as a runtime method.

5.2.2 Performance of the ANN Stress Model with Hand-crafted Feature Extraction. Now we test the ANN stress model with hand-crafted feature extraction, whose structure is demonstrated in Section 4.4.

The stress estimation accuracy results of this ANN stress model on 3D ICs with uniform TSV distribution, nonuniform TSV distribution, and random TSV distribution are collected in Table 6, Table 7, and Table 8, respectively. By comparing the accuracy results in Tables 3, 4, and 5 with Tables 6, 7, and 8, we see that the ANN stress model with hand-crafted feature extraction is generally more accurate than the normal ANN-based stress model. Specifically, the ANN stress model with hand-crafted feature extraction reaches its best accuracy with two hidden layers configured

Table 11. Stress Estimation Accuracy Results of the CNN-based Stress Models on 3D IC with Nonuniform TSV Distribution in Figure 11

Convolutional layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
5 × 5, 3	3.51	4.75	4.23	20.1	17.3	16.5	3.48	4.70	4.27	18.5	15.8	17.3
5 × 5, 6	3.36	3.61	3.56	16.5	19.7	13.8	3.39	3.64	3.55	17.3	12.1	13.5
5 × 5, 10	2.78	3.01	2.99	13.7	12.9	9.93	2.67	2.98	3.12	12.0	13.4	10.9
5 × 5, 15	2.80	3.85	3.67	13.0	17.1	14.6	2.73	3.79	3.71	13.2	12.7	14.7
5 × 5, 3; 5 × 5, 6	2.57	3.23	2.97	9.36	13.5	14.0	2.63	3.36	2.86	15.3	14.1	12.0
5 × 5, 6; 5 × 5, 15	2.43	2.69	2.56	17.1	12.4	9.43	2.30	2.72	2.74	14.2	12.3	9.91
5 × 5, 10; 5 × 5, 25	2.16	1.83	1.91	8.94	9.55	10.7	2.11	1.78	1.96	10.1	11.7	9.62
5 × 5, 15; 5 × 5, 30	2.79	2.56	2.55	9.76	14.8	12.4	2.68	3.53	2.44	12.6	13.1	9.01
3 × 3, 3	2.60	3.85	3.43	13.3	17.2	11.7	2.53	3.81	3.58	12.1	12.7	11.3
3 × 3, 6	2.89	3.57	3.10	14.1	13.8	11.6	2.76	3.49	3.22	11.3	14.6	11.4
3 × 3, 10	2.99	3.02	2.78	9.93	14.2	11.9	2.82	3.18	2.70	14.1	9.89	16.1
3 × 3, 15	2.29	3.95	2.86	8.66	14.2	13.9	2.69	4.06	3.75	13.9	18.1	13.6
3 × 3, 3; 3 × 3, 6	2.50	4.11	3.82	9.15	14.6	17.9	2.46	3.78	3.95	12.8	13.7	14.2
3 × 3, 10; 3 × 3, 25	2.33	2.73	2.58	13.2	11.2	9.38	2.35	2.71	2.61	12.1	14.9	13.4
3 × 3, 15; 3 × 3, 30	2.50	3.64	2.70	9.46	14.0	15.8	2.48	3.52	2.73	10.9	14.6	15.7
3 × 3, 3; 3 × 3, 6; 3 × 3, 10	2.79	3.92	3.20	12.1	13.9	12.7	2.86	3.66	3.34	15.3	14.3	11.6
3 × 3, 6; 3 × 3, 15; 3 × 3, 20	2.93	3.54	2.81	10.6	11.8	16.4	2.85	3.97	2.93	14.7	16.5	13.9

The shared configurations of all CNN stress models are given in Table 9. The configuration parameters in different convolutional layers are separated by semicolon in the "Convolutional layer configuration" column, with "$i \times i$, j" meaning the kernel size is $i \times i$ and the convolution number working in parallel is j in the corresponding layer; 300K, 323K, and 548K denote the stress-free temperatures.

as "55; 15." Its average error is within 5% for all test cases, which is much smaller than that of the most accurate normal ANN stress model (also with two hidden layers configured as "55; 15"). This is because the rotational symmetry induced redundancy in the input of the normal ANN stress model is eliminated by using the hand-crafted feature extraction scheme presented in Section 4.4.

The computing time of the ANN stress model with hand-crafted feature extraction is essentially the same as the normal ANN stress model by analyzing the data in Table 13. This is because the two ANN stress models basically have the same internal fully connected structure.

5.2.3 Performance of the CNN-based Stress Model. In this part, we test the performance of the CNN-based stress model.

Since the CNN-based stress model has complex structure, it has a large number of configuration combinations. To better demonstrate the accuracy of CNN stress models with different sizes, we first perform a lot of experiments to determine some common configuration settings that work well for all CNN stress models, regardless of the convolutional layer configuration. These common configuration settings are shown in Table 9, including pooling type, pooling window size, and neuron number in hidden layer of the fully connected ANN.

With the shared configuration settings, we created several CNN stress models by changing the convolution layer number,[2] convolution kernel size, and the convolution number working in parallel in each convolution layer. The accuracy performances of these CNN stress models are given

[2]The pooling layer number should also be changed accordingly, because each convolutional layer should be followed by a pooling layer.

Table 12. Stress Estimation Accuracy Results of the CNN-based Stress Models on 3D IC with Random TSV Distribution in Figure 12

Convolutional layer configuration	Synthetic power						SPEC power					
	Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
	300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
5 × 5, 3	3.57	4.62	4.18	21.5	16.8	17.3	3.56	4.86	4.37	17.4	16.9	16.2
5 × 5, 6	3.42	3.58	3.63	17.9	18.2	15.0	3.51	3.69	3.52	17.9	12.3	17.5
5 × 5, 10	2.69	3.21	2.82	12.9	13.1	9.96	2.73	2.84	3.31	12.8	17.1	12.5
5 × 5, 15	2.76	3.92	3.34	12.8	16.3	15.6	2.74	3.86	3.82	13.9	14.5	16.4
5 × 5, 3; 5 × 5, 6	2.73	3.41	2.88	9.43	12.8	13.6	2.73	3.42	2.85	14.9	13.2	15.0
5 × 5, 6; 5 × 5, 15	2.48	2.73	2.49	14.6	14.4	9.83	2.37	2.76	2.65	15.2	13.6	9.98
5 × 5, 10; 5 × 5, 25	2.14	1.89	1.97	12.5	9.88	11.4	2.08	1.89	1.98	10.9	13.5	10.4
5 × 5, 15; 5 × 5, 30	2.83	2.76	2.39	9.89	12.8	13.6	2.58	3.56	2.72	11.9	16.5	10.8
3 × 3, 3	2.72	3.38	3.71	11.9	14.2	14.8	2.46	3.93	3.76	12.8	11.5	12.9
3 × 3, 6	2.68	3.42	3.15	13.6	14.9	10.8	2.63	3.58	3.24	11.7	13.2	12.4
3 × 3, 10	2.96	3.14	2.66	10.5	13.8	15.7	2.79	3.32	2.83	13.6	12.7	14.5
3 × 3, 15	2.42	3.86	2.73	9.76	13.2	12.7	3.10	4.22	3.68	15.2	17.3	14.8
3 × 3, 3; 3 × 3, 6	2.56	4.23	3.71	9.15	14.9	15.8	2.53	3.66	3.87	11.9	12.6	13.4
3 × 3, 10; 3 × 3, 25	2.46	2.59	2.62	14.1	12.8	9.86	2.42	2.59	2.74	14.5	15.2	14.7
3 × 3, 15; 3 × 3, 30	2.58	3.54	2.65	9.83	12.1	16.3	2.58	3.46	2.86	12.1	13.2	14.8
3 × 3, 3; 3 × 3, 6; 3 × 3, 10	2.88	3.95	3.41	13.2	11.5	13.8	2.96	3.53	3.30	14.1	16.2	12.3
3 × 3, 6; 3 × 3, 15; 3 × 3, 20	2.87	3.63	2.89	10.9	14.1	15.2	2.76	3.67	2.98	12.5	14.1	12.8

The shared configurations of all CNN stress models are given in Table 9. The configuration parameters in different convolutional layers are separated by semicolon in the "Convolutional layer configuration" column, with "$i \times i$, j" meaning the kernel size is $i \times i$ and the convolution number working in parallel is j in the corresponding layer; 300K, 323K, and 548K denote the stress-free temperatures.

in Table 10, Table 11, and Table 12 for 3D ICs with uniform TSV distribution, nonuniform TSV distribution, and random TSV distribution, respectively. The configuration parameters in different convolutional layers are separated by semicolon in the "Convolutional layer configuration" column, with "$i \times i$, j" meaning the kernel size is $i \times i$ and the parallel convolution number is j in the corresponding layer. Although we have tested many CNN stress models with different convolutional layer configurations, we only show the ones specially chosen to represent both structure diversity and error trends in the table, also with the best one we found included.

By comparing the accuracy results of CNN stress model (Tables 10, 11, and 12) with those of the normal ANN-based stress model (Tables 3, 4, and 5) and ANN stress model with hand-crafted feature extraction (Tables 6, 7, and 8), we can see that the CNN stress model outperforms both the normal ANN-based stress model and the ANN stress model with hand-crafted feature extraction in stress estimation accuracy. Specifically, the CNN stress model reaches its best accuracy with the convolutional layer configuration "5 × 5, 10; 5 × 5, 25," which is also the most accurate ANN stress model we have tested in the experiment. It achieves an average error to be within 2.2% for all test cases. This high accuracy is achieved thanks to the excellent feature extraction ability of the CNN structure.

The computing speed of the CNN-based stress model is also very fast according to Table 13. With the same accuracy, stress estimation using the CNN-based stress model is significantly faster than the other two kinds of ANN stress models. For example, the CNN stress model with configuration "3 × 3, 3" is more accurate than the ANN stress model with hand-crafted feature extraction with configuration "55; 15," but the former stress model has a smaller computing overhead (0.07ms) than

Table 13. Runtime Results of the Three ANN-based Stress Models

Normal ANN		ANN with extraction		CNN	
Hidden layer configuration	Time (ms)	Hidden layer configuration	Time (ms)	Convolutional layer configuration	Time (ms)
20	0.07	20	0.07	5×5, 3	0.09
40	0.11	40	0.11	5×5, 6	0.16
60	0.14	60	0.13	5×5, 10	0.23
140	0.29	140	0.32	5×5, 15	0.43
200	0.39	200	0.41	5×5, 3; 5×5, 6	0.15
40; 10	0.12	40; 10	0.10	5×5, 6; 5×5, 15	0.24
55; 15	0.17	55; 15	0.13	5×5, 10; 5×5, 25	0.38
70; 20	0.19	70; 20	0.15	5×5, 15; 5×5, 30	0.45
90; 40	0.25	90; 40	0.19	3×3, 3	0.07
140; 60	0.33	140; 60	0.35	3×3, 6	0.12
200; 80	0.41	200; 80	0.43	3×3, 10	0.22
40; 10; 5	0.13	40; 10; 5	0.11	3×3, 15	0.34
55; 15; 10	0.15	55; 15; 10	0.17	3×3, 3; 3×3, 6	0.09
80; 30; 15	0.20	80; 30; 15	0.19	3×3, 10; 3×3, 25	0.22
120; 60; 20	0.26	120; 60; 20	0.23	3×3, 15; 3×3, 30	0.40
200; 80; 30	0.40	200; 80; 30	0.47	3×3, 3; 3×3, 6; 3×3, 10	0.10
60; 30; 15; 5	0.16	60; 30; 15; 5	0.14	3×3, 6; 3×3, 15; 3×3, 20	0.17
200; 120; 60; 20	0.49	200; 120; 60; 20	0.46		

"Normal ANN" denotes for normal ANN-based stress model, "ANN with extraction" means ANN stress model with hand-crafted feature extraction, and "CNN" represents CNN-based stress model.

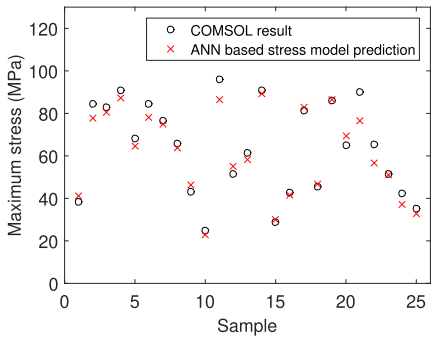
the later stress model (0.13ms). This is because the convolution and pooling operation shrinks the network size drastically by removing the redundancies, which greatly improves the computing efficiency without sacrificing accuracy.

We also compare the computing speed of the CNN-based stress model with COMSOL FEM. In our experiment, FEM spends around 10s to generate each sample, while the most accurate CNN stress model (with configuration $5 \times 5, 10; 5 \times 5, 25$) uses only 0.38ms. So the CNN stress model gains around 2.6×10^4 speed up against FEM.

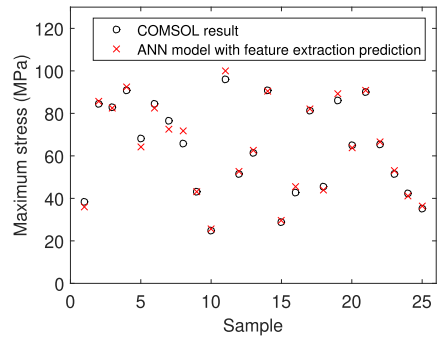
5.3 Comparison of the Three ANN Stress Models

To see a clearer accuracy comparison, we plot the stress estimation results of three ANN stress models in Figure 13, where each model has the configuration that achieves the best accuracy in the previous accuracy tests for its kind. Although all models are able to provide acceptable stress estimation results, the CNN stress model clearly shows superior results by providing nearly identical maximum stress estimations as the golden COMSOL results.

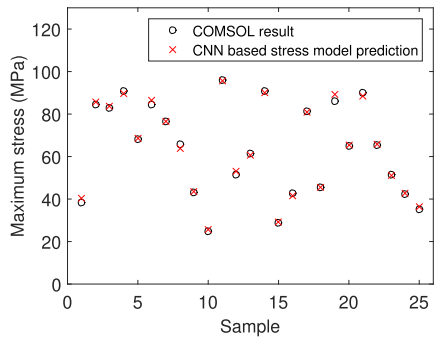
To take both accuracy and runtime into consideration, we also plot the average error versus runtime comparison results in Figure 14, by changing the layer sizes and layer numbers for each kind of stress model. We can see that the CNN stress model performs best for all stress-free temperature settings, because it has higher accuracy than the other two models for the same runtime. It is also observed that the ANN stress model with hand-crafted extraction performs better than the normal ANN stress model in this joint accuracy-runtime comparison.



(a) Stress estimation results of the normal ANN based stress model with configuration "55; 15".



(b) Stress estimation results of the ANN stress model with hand-crafted feature extraction with configuration "55; 15".



(c) Stress estimation results of the CNN stress model with configuration "5×5, 10; 5×5, 25".

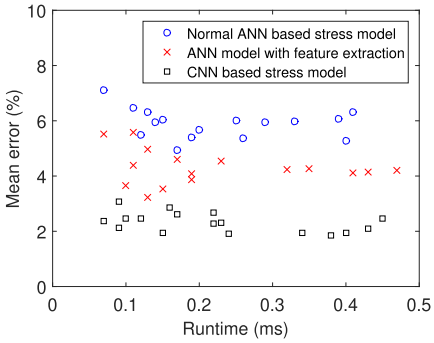
Fig. 13. Stress estimation results comparison of three ANN-based stress models on 25 randomly selected validation samples. The stress-free temperature is 323K.

5.4 Comparison with the Traditional Stress Estimation Method

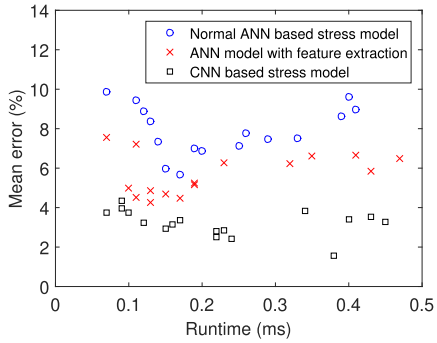
Finally, to show the advantage of the new ANN-based runtime stress estimation method, we compare it with the traditional stress estimation method.

Although FEM are widely used to estimate the stress distribution of 3D ICs at design time [6, 16, 22, 25, 30, 40, 42], they are computationally too expensive to be used at runtime. Instead, analytical stress estimation methods were proposed for fast stress estimation [2, 23] by computing stress at a uniform temperature distribution. Most of these analytical stress estimation methods are designed to perform corner-based analysis, which estimate the worst-case stress with a fixed uniform temperature distribution (usually use room temperature for the high stress-free temperature case [23]). In other words, they are not designed to deal with the changing temperature and nonuniform temperature distribution in 3D IC, which can be fully considered in the new ANN-based stress estimation method. Still, we find the method in Reference [23] is at least able to estimate stress with temperature as a variable (but still uniform temperature distribution assumed), so we compare it with the new ANN-based stress estimation method.

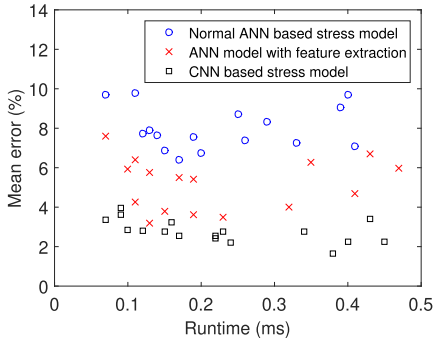
For the comparison settings, since the traditional method [23] cannot deal with nonuniform temperature distribution, we use the average temperature around each TSV as its temperature



(a) Joint accuracy-runtime comparison with stress-free temperature 300 K.



(b) Joint accuracy-runtime comparison with stress-free temperature 323 K.



(c) Joint accuracy-runtime comparison with stress-free temperature 548 K.

Fig. 14. Joint accuracy–runtime comparison of the three ANN stress models by changing the model size for each kind of stress model. The CNN stress model has the best accuracy for the same runtime.

Table 14. Accuracy Comparison between the CNN-based Stress Model ("CNN" in the Table) and the Traditional Stress Estimation Method [23]

TSV	Power	Traditional method [23]						CNN (5×5, 10; 5×5, 25)					
		Mean error (%)			Max error (%)			Mean error (%)			Max error (%)		
		300K	323K	548K	300K	323K	548K	300K	323K	548K	300K	323K	548K
Uniform	Synthetic	12.8	10.9	13.4	76.3	86.6	123	1.85	1.58	1.64	7.38	7.97	8.53
Uniform	SPEC	12.6	10.7	13.6	94.0	92.1	97.1	1.91	1.74	1.83	9.31	10.8	8.72
Nonuniform	Synthetic	12.8	11.2	13.3	117	103	114	2.16	1.83	1.91	8.94	9.55	10.7
Nonuniform	SPEC	11.9	10.8	13.5	87.6	98.5	106	2.11	1.78	1.96	10.1	11.7	9.62

input. For our method, we pick the CNN-based stress model with convolutional layer configuration "5×5, 10; 5×5, 25" to take part in the comparison.

The accuracy comparison results are collected in Table 14. The CNN-based stress model clearly shows better accuracy in all tests, with around six times smaller average error compared with the traditional method. This is because our new stress model is able to consider temperature distribution/gradient around each TSV. In contrast, the existing method assumes the temperature

distribution around each TSV to be uniform, which is okay for the worst-case corner-based analysis but introduces large error for runtime stress estimation.

The accuracy advantage of our ANN-based stress estimation method does not come free. The ANN-based stress estimation method includes neural network computing with temperatures around each TSV as input, meaning it has larger computing overhead (but still very small: only 0.38ms as shown in Table 13) compared to the traditional method [23], which has nearly no computing overhead at all. However, we believe the accuracy advantage of the new method is large enough to overweigh its disadvantage in computing overhead.

6 CONCLUSION

In this article, we have proposed a runtime stress estimation method for reliability management of 3D IC using the ANN stress model, which estimates the important stress information for reliability management with temperature around TSV as input. Three ANN-based stress models with three major types of ANNs are analyzed for fast stress estimation, including the normal ANN-based stress model, the ANN stress model with hand-crafted feature extraction, and the CNN-based stress model. The structure of each ANN stress model is presented and the principles of the ANN-based stress estimation are analyzed. Experiments with different configurations of ANN stress models show that the new method is able to estimate important stress information at extremely fast speed with good accuracy for runtime 3D IC reliability management usage. Among the three ANN-based stress models, the CNN-based stress model has the best performance considering both stress estimation accuracy and computing overhead. Comparison with traditional method reveals that the new ANN-based stress estimation method is much more accurate with a slightly larger but still very small computing overhead.

REFERENCES

[1] JEDEC Solid State Technology Association. 2003. *Failure Mechanisms and Models for Semiconductor Devices*. Technical Report JEP122H.

[2] Krit Athikulwongse, Jae-Seok Yang, David Z. Pan, and Sung Kyu Lim. 2013. Impact of mechanical stress on the full chip timing for through-silicon-via-based 3-D ICs. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.* 32, 6 (2013), 905–917.

[3] Kaustav Banerjee, Shukri J. Souri, Pawan Kapur, and Krishna C. Saraswat. 2001. 3-D ICs: A novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration. *Proc. IEEE* 89, 5 (May 2001), 602–633.

[4] A. S. Budiman, H.-A.-S. Shin, B.-J. Kim, S.-H. Hwang, H.-Y. Son, M.-S. Suh, Q.-H. Chung, K.-Y. Byun, N. Tamura, M. Kunz, and Y.-C. Joo. 2012. Measurement of stresses in Cu and Si around through-silicon via by synchrotron X-ray microdiffraction for 3-dimensional integrated circuits. *Microelectr. Reliabil.* 52, 3 (2012), 530–533.

[5] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. [n.d.]. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. CoRR abs/1512.01274. Retrieved from http://arxiv.org/abs/1512.01274

[6] Tao Chen, Dunming Liao, and Jianxin Zhou. 2013. Numerical simulation of casting thermal stress and deformation based on finite difference method. *Materials Science Forum (MSF)* 762 (2013), 224–229.

[7] James Donald and Margaret Martonosi. 2006. Techniques for multicore thermal management: Classification and new exploration. In *Proceedings of the International Symposium on Computer Architecture (ISCA'06)*. 78–88.

[8] Wei-Ping Dow, Chun-Weia Lu, Jing-Yuan Lin, and Fu-Chiang Hsu. 2011. Highly selective Cu electrodeposition for filling through silicon holes. *Electrochem. Solid-State Lett.* 14, 6 (2011), 63–67.

[9] Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, and Doug Burger. 2012. Dark silicon and the end of multicore scaling. *IEEE MICRO* 32, 3 (May 2012), 122–134.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. The MIT Press.

[11] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. In *Proceedings of the International Conference on Machine Learning (ICML'13)*. 1319–1327.

[12] John L. Henning. 2000. SPEC CPU 2000: Measuring CPU performance in the new millennium. *IEEE Comput.* 1, 7 (July 2000), 28–35.

[13] John L. Henning. 2006. SPEC CPU2006 benchmark descriptions. *ACM SIGARCH Comput. Arch. News* 34, 4 (Sep. 2006), 1–17.

[14] Tengfei Jiang, Suk-Kyu Ryu, Qiu Zhao, Jay Im, Rui Huang, and Paul S. Ho. 2013. Measurement and analysis of thermal stresses in 3D integrated structures containing through-silicon-vias. *Microelectr. Reliabil.* 53, 1 (2013), 53–62.

[15] Moongon Jung, Joydeep Mitra, David Z. Pan, and Sung Kyu Lim. 2014. TSV stress-aware full-chip mechanical reliability analysis and optimization for 3D IC. *Commun. ACM* 57, 1 (Jan. 2014), 107–115.

[16] Moongon Jung, David Z. Pan, and Sung Kyu Lim. 2012. Chip/package co-analysis of thermo-mechanical stress and reliability in TSV-based 3D ICs. In *Proceedings of the Design Automation Conference (DAC'12)*. IEEE, 317–326.

[17] John H. Lau and Tang Gong Yue. 2009. Thermal management of 3D IC integration with TSV (through silicon via). In *Proceedings of the Electronic Components and Technology Conference (ECTC'09)*. 635–640.

[18] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neur. Comput.* 1, 4 (1989), 541–551.

[19] Feihui Li, Chrysostomos Nicopoulos, Thomas Richardson, Yuan Xie, Vijaykrishnan Narayanan, and Mahmut Kandemir. 2006. Design and management of 3D chip multiprocessors using network-in-memory. In *Proceedings of the International Symposium on Computer Architecture (ISCA'06)*. 130–141.

[20] Xi Liu, Qiao Chen, Venkatesh Sundaram, Rao R. Tummala, and Suresh K. Sitaraman. 2013. Failure analysis of through-silicon vias in free-standing wafer under thermal-shock test. *Microelectr. Reliabil.* 53, 1 (2013), 70–78.

[21] Jieyi Long, Seda Ogrenci Memik, Gokhan Memik, and Rajarshi Mukherjee. 2008. Thermal monitoring mechanisms for chip multiprocessors. *ACM Trans. Arch. Code Opt.* 5, 2 (Aug. 2008), 9:1–9:33.

[22] Kuan H. Lu, Xuefeng Zhang, Suk-Kyu Ryu, Jay Im, Rui Huang, and Paul S. Ho. 2009. Thermo-mechanical reliability of 3-D ICs containing through silicon vias. In *Proceedings of the Electronic Components and Technology Conference (ECTC'09)*. 630–634.

[23] Sravan K. Marella and Sachin S. Sapatnekar. 2015. A holistic analysis of circuit performance variations in 3-D ICs with thermal and TSV-induced stress considerations. *IEEE Trans. VLSI Syst.* 23, 7 (Jul. 2015), 1308–1321.

[24] Jie Meng, Katsutoshi Kawakami, and Ayse K Coskun. 2012. Optimizing energy efficiency of 3-D multicore systems with stacked DRAM under power and thermal constraints. In *Proceedings of the Design Automation Conference (DAC'12)*. 648–655.

[25] Joydeep Mitra, Moongon Jung, Suk-Kyu Ryu, Rui Huang, Sung-Kyu Lim, and David Z. Pan. 2011. A fast simulation framework for full-chip thermo-mechanical stress and reliability analysis of through-silicon-via based 3D ICs. In *Proceedings of the Electronic Components and Technology Conference (ECTC'11)*. IEEE, 746–753.

[26] Santiago Pagani, Heba Khdr, Jian-Jia Chen, Muhammad Shafique, Minming Li, and Jörg Henkel. 2017. Thermal safe power (TSP): Efficient power budgeting for heterogeneous manycore systems in dark silicon. *IEEE Trans. Comput.* 66, 1 (Jan. 2017), 147–162.

[27] David Z. Pan, Sung Kyu Lim, Krit Athikulwongse, Moongon Jung, Joydeep Mitra, Jiwoo Pak, Mohit Pathak, and Jae-Seok Yang. 2012. Design for manufacturability and reliability for TSV-based 3D ICs. In *Proceedings of the Asia South Pacific Design Automation Conference (ASP-DAC'12)*. 750–755.

[28] Robert S. Patti. 2006. Three-dimensional integrated circuits and the future of system-on-chip designs. *Proc. IEEE* 94, 6 (Jun. 2006), 1214–1224.

[29] P. Saettler, M. Hecker, M. Boettcher, C. Rudolph, and K.J. Wolter. 2015. μ-Raman spectroscopy and FE-modeling for TSV-stress-characterization. *Microelectr. Eng.* 137 (2015), 105–110.

[30] Anderson Pires Singulani, Hajdin Ceric, and Siegfried Selberherr. 2013. Stress evolution in the metal layers of TSVs with Bosch scallops. *Microelectr. Reliabil.* 53, 9 (2013), 1602–1605.

[31] Arvind Sridhar, Alessandro Vincenzi, Martino Ruggiero, Thomas Brunschwiler, and David Atienza. 2010. 3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling. In *Proceedings of the International Conference on Computer Aided Design (ICCAD'10)*. 463–470.

[32] Jayanth Srinivasan. 2006. *Lifetime Reliability Aware Microprocessors*. Ph.D. Dissertation. University of Illinois at Urbana-Champaign.

[33] Jayanth Srinivasan, Sarita V. Adve, Pradip Bose, and Jude A. Rivers. 2004. The case for lifetime reliability-aware microprocessors. In *Proceedings of the International Symposium on Computer Architecture (ISCA'04)*. 276–287.

[34] Hai Wang, Jian Ma, Sheldon X.-D. Tan, Chi Zhang, He Tang, Keheng Huang, and Zhenghong Zhang. 2016. Hierarchical dynamic thermal management method for high-performance many-core microprocessors. *ACM Trans. Des. Autom. Electr. Syst.* 22, 1 (Jul. 2016), 1:1–1:21.

[35] Hai Wang, Diya Tang, Ming Zhang, Sheldon X.-D. Tan, Chi Zhang, He Tang, and Yuan Yuan. 2019. GDP: A greedy based dynamic power budgeting method for multi/many-core systems in dark silicon. *IEEE Trans. Comput.* 68, 4 (Apr. 2019), 526–541.

[36] Hai Wang, Jiachun Wan, Sheldon X.-D. Tan, Chi Zhang, He Tang, Yuan Yuan, Keheng Huang, and Zhenghong Zhang. 2018. A fast leakage-aware full-chip transient thermal estimation method. *IEEE Trans. Comput.* 67, 5 (May 2018), 617–630.

[37] Roshan Weerasekera, Li-Rong Zheng, Dinesh Pamunuwa, and Hannu Tenhunen. 2007. Extending systems-on-chip to the third dimension: Performance, cost and technological tradeoffs. In *Proceedings of the International Conference on Computer Aided Design (ICCAD'07)*. 212–219.

[38] Jae-Seok Yang, Krit Athikulwongse, Young-Joon Lee, Sung Kyu Lim, and David Z. Pan. 2010. TSV stress aware timing analysis with applications to 3D-IC layout optimization. In *Proceedings of the Design Automation Conference (DAC'10)*. 803–806.

[39] Song Yao, Xiaoming Chen, Yu Wang, Yuchun Ma, Yuan Xie, and Huazhong Yang. 2014. Efficient region-aware P/G TSV planning for 3D ICs. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED'14)*. 171–178.

[40] Li Yu, Wen-Yao Chang, Kewei Zuo, Jean Wang, Douglas Yu, and Duane Boning. 2012. Methodology for analysis of TSV stress induced transistor variation and circuit performance. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED'12)*. 216–222.

[41] Lang Zhang, Hai Wang, and Sheldon X.-D. Tan. 2016. Fast stress analysis for runtime reliability enhancement of 3D IC using artificial neural network. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED'16)*. 173–178.

[42] Hao Zhou, Hengliang Zhu, Tao Cui, David Z. Pan, Dian Zhou, and Xuan Zeng. 2018. Thermal stress and reliability analysis of TSV-based 3-D ICs with a novel adaptive strategy finite element method. *IEEE Trans. VLSI Syst.* 26, 7 (2018), 1312–1325.

[43] Qiaosha Zou, Eren Kursun, and Yuan Xie. 2017. Thermomechanical stress-aware management for 3-D IC designs. *IEEE Trans. VLSI Syst.* 25, 9 (2017), 2678–2682.

Received November 2018; revised July 2019; accepted August 2019