Solder joint reliability risk estimation by AI-assisted simulation framework with genetic algorithm to optimize the initial parameters for AI models

Yuan, Cadmus ; Fan, Xuejun; Zhang, Gouqi

DOI
10.3390/ma14174835

Publication date
2021

Document Version
Final published version

Published in
Materials

Citation (APA)
Yuan, C., Fan, X., & Zhang, G. (2021). Solder joint reliability risk estimation by AI-assisted simulation framework with genetic algorithm to optimize the initial parameters for AI models. *Materials*, *14*(17), 1-19. Article 4835. https://doi.org/10.3390/ma14174835

materials

MDPI

Article

Solder Joint Reliability Risk Estimation by AI-Assisted Simulation Framework with Genetic Algorithm to Optimize the Initial Parameters for AI Models

Cadmus Yuan [1,*], Xuejun Fan [2] and Gouqi Zhang [3]

[1] Department of Mechanical and Computer-Aided Engineering, Feng Chia University, Taichung 407082, Taiwan
[2] Department of Mechanical Engineering, Lamar University, Beaumont, TX 77710, USA; xuejun.fan@lamar.edu
[3] Department of Electronic Components, Technology, and Materials, Delft University of Technology, 2628 CD Delft, The Netherlands; g.q.zhang@tudelft.nl
[*] Correspondence: cayuan@fcu.edu.tw; Tel.: +886-939-873-055

Abstract: Solder joint fatigue is one of the critical failure modes in ball-grid array packaging. Because the reliability test is time-consuming and geometrical/material nonlinearities are required for the physics-driven model, the AI-assisted simulation framework is developed to establish the risk estimation capability against the design and process parameters. Due to the time-dependent and nonlinear characteristics of the solder joint fatigue failure, this research follows the AI-assisted simulation framework and builds the non-sequential artificial neural network (ANN) and sequential recurrent neural network (RNN) architectures. Both are investigated to understand their capability of abstracting the time-dependent solder joint fatigue knowledge from the dataset. Moreover, this research applies the genetic algorithm (GA) optimization to decrease the influence of the initial guessings, including the weightings and bias of the neural network architectures. In this research, two GA optimizers are developed, including the “back-to-original” and “progressing” ones. Moreover, we apply the principal component analysis (PCA) to the GA optimization results to obtain the PCA gene. The prediction error of all neural network models is within 0.15% under GA optimized PCA gene. There is no clear statistical evidence that RNN is better than ANN in the wafer level chip-scaled packaging (WLCSP) solder joint reliability risk estimation when the GA optimizer is applied to minimize the impact of the initial AI model. Hence, a stable optimization with a broad design domain can be realized by an ANN model with a faster training speed than RNN, even though solder fatigue is a time-dependent mechanical behavior.

Keywords: solder joint fatigue risk estimation; wafer level chip-scaled packaging; artificial neural network; recurrent neural network; generic algorithm; principle component analysis; time/temperature-dependent nonlinearity

check for updates

Citation: Yuan, C.; Fan, X.; Zhang, G. Solder Joint Reliability Risk Estimation by AI-Assisted Simulation Framework with Genetic Algorithm to Optimize the Initial Parameters for AI Models. *Materials* **2021**, *14*, 4835. https://doi.org/10.3390/ma14174835

Academic Editor: Kuo-Ning Chiang

Received: 15 July 2021
Accepted: 23 August 2021
Published: 26 August 2021

1. Introduction

Solder joint reliability is one of the most critical issues for most ball-grid array packaging types. The time dependency of this failure mechanism requires considerable experiment time to obtain statistically reliable results. On the other hand, the nonlinear material/geometry properties are required for the finite element (FE) modeling to retrieve trustable results, which can be validated by the experimental results. Hence, both the reliability experiment and numerical modeling require unique expertise to conduct the relevant tasks, which creates a technical barrier for the design for reliability.

The neural network (NN)-based AI algorithms were applied to assist the design and simulation of the solder joint risk assessment. Chou and Chiang [1] and Hsiao and Chiang [2] developed an AI-assisted design and simulation framework. It includes the virtual prototyping of solder fatigue failure mode with the geometrical/material nonlinearity and the proper validation by the experimental results. The training database is generated from the parametric FE model. Next, AI modeling is trained by the selected data points from the database and validated by the rest. Careful validation works should be conducted to prove the representation capability of the AI model to the FE dataset.

The NN approaches of AI modeling received more and more attention due to the capability of abstracting the knowledge from the database without the pre-defined framework nor prior knowledge/experience. ANN is a basic architecture of NN with weak representation capability for sequential events. However, the solder joint fatigue mechanism is highly time-dependent. Accordingly, the sequential NN techniques, including the recurrent neural network (RNN), GRU, and LSTM, are successfully applied to the time-dependent failure mechanism for electronic packaging. However, these sequential NN methods consider the recurrent parameters and iterations that induce learning difficulties and require considerable computation resources. Yuan and Lee [3] applied the sequential NN to model the time-dependent nature of the solder joint fatigue, and the average error norms below 1.213×10^{-4} were achieved. Yuan et al. [4] developed a gated neural network technique to learn the performance shifting of the solid-state lighting (SSL) lamp over time. Meszmer et al. [5] applied many NN techniques to study which is the best for the electronic packaging, and the sequential NN performed best, including the gate recurrent unit (GRU) and long short-term memory (LSTM) architectures because of their capability to capture the characteristics of the sequential dataset. Selvanayagam et al. [6] applied the AI-assisted modeling concept for the improvement of the packaging warpage. Tabaza et al. [7] applied the non-sequential NN to simulate the time-dependent hysteretic response of a viscoelastic material. The possibility of using the non-sequential NN, such as NN, upon a time-dependent engineering problem remains a challenge.

Considering the learning procedure, NN is a parametric AI modeling method. In addition to the network structure, NN utilizes the parameter, including the weightings and bias, to learn the knowledge from the database. However, NN requires the initial guessing of parameters to start the learning process, and the improper selection of the initial parameter results in a slow convergence speed and bumpy learning procedure. The genetic algorithm (GA) is always applied. A genetic algorithm (GA) is an optimization method proposed by John Holland to find the approximate solutions. This algorithm is a specific form of an evolutionary algorithm in which evolutionary biology techniques such as inheritance and mutation are used. In genetic algorithms, to obtain the optimal solution, the appropriate responses of a generation are combined based on the principle of the survival of the fittest environment [8]. White and Ligomenides proposed a topology and weighting optimization algorithm for neural networks [9]. Juang [10] hybrids the GA and particle swarm optimization (PSO). The new generation of the GA can be generated not only by the crossover and mutation but also by PSO. Ding et al. [11], Ahmadizar et al. [12] and Arabasadi et al. [13] apply the GA for weighting optimization of NN. However, few literature had covered GA for the sequential network nor utilized the continuously evolving nature of the NN backpropagation.

Based on the AI-assisted simulation framework, this research investigates the possibility of using the ANN instead of the RNN in physically sequential issues, such as the solder joint fatigue mechanisms. Secondly, this research develops a genetic algorithm method to obtain the optimal parameters for NN learning.

This paper is organized as follows: the fundamental scientific issues and the literature review are described in the Introduction. The following section, Theory, provides the basic theoretic approaches that have been applied in this research. The execution of the AI-assisted simulation framework is explained in the sections The AI-assisted simulation framework and FE dataset preparation and The design of the AI modeling. The AI model training with GA optimized initial parameters section summarizes the learning experience of the AI modeling. The conclusion of this paper is given in the last section.

2. Theory

In GA, the fitness criterion is first defined to quantify the members of the current generation with more compatibility are more likely to generate the next population [8,9]. The fitness criterion (F) is set as follows:

$$F = 1/r \tag{1}$$

where the r is the error norm of learning. The members with higher F values are more likely to generate the next population by the crossover and mutation operators.

The member is also called the chromosome, which is made up of many genes. The gene is constructed by many base-pairs (bps) [12]. Given a pair of parent chromosomes with m genes, the crossover operator will generate 2^m different offsprings by the recombination of the genes that are from the parent [13]. To cover the genetic representation ultimately, the whole 2^m offsprings are forming to the next generation without any possibility. As illustrated in Figure 1, there is one pair of parents with three genes. Eight ($= 2^3$) offsprings are generated by the recombination of the parents' gene, which is the definition of the crossover operator.

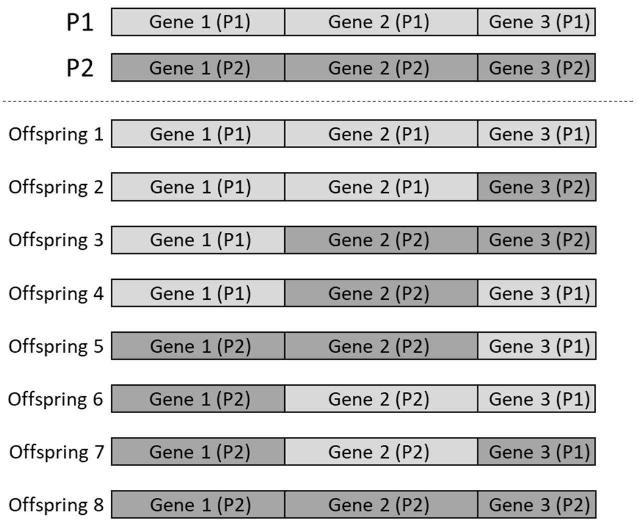
P1 Gene 1 (P1) Gene 2 (P1) Gene 3 (P1)
P2 Gene 1 (P2) Gene 2 (P2) Gene 3 (P2)
Offspring 1 Gene 1 (P1) Gene 2 (P1) Gene 3 (P1)
Offspring 2 Gene 1 (P1) Gene 2 (P1) Gene 3 (P2)
Offspring 3 Gene 1 (P1) Gene 2 (P2) Gene 3 (P2)
Offspring 4 Gene 1 (P1) Gene 2 (P2) Gene 3 (P1)
Offspring 5 Gene 1 (P2) Gene 2 (P2) Gene 3 (P1)
Offspring 6 Gene 1 (P2) Gene 2 (P2) Gene 3 (P1)
Offspring 7 Gene 1 (P2) Gene 2 (P2) Gene 3 (P1)
Offspring 8 Gene 1 (P2) Gene 2 (P2) Gene 3 (P2)

Figure 1. The illustration of the crossover operator.

Therefore, if the top n chromosomes are selected, there are $\frac{n(n-1)}{2}$ parent combinations and there are maximum $\frac{n(n-1)}{2} \times 2^m$ offsprings possibilities. However, if the parent chromosomes consisted of many similar genes, it might induce many duplicated offsprings. These offsprings with the same genes will be removed to save the computation resource.

The mutation operator [13] is used to make changes in the genes of a member of the current generation to produce a new member. The mutation occurs at the bps level and is controlled by the mutation rate [12]. When the mutation is invoked at certain bps, the representation bps will be replaced by the opposite parent bps. For example, as illustrated in Figures 1 and 2, the 9th bps of the first gene of offspring 7 will mutate, and the original bps will be replaced by the 9th bps of the first gene of parent 1 (P1).

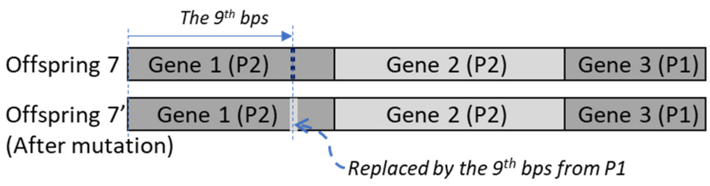
The 9th bps
Offspring 7 Gene 1 (P2) Gene 2 (P2) Gene 3 (P1)
Offspring 7' (After mutation) Gene 1 (P2) Gene 2 (P2) Gene 3 (P1)
Replaced by the 9th bps from P1

Figure 2. The illustration of the crossover and mutation operator.

After few completed genetic algorithm iterations, these best chromosomes of each iteration are not the same due to the nature of the neural network. The principal component analysis (PCA) algorithm is applied to build a super chromosome based on these best chromosomes. If the complete chromosome, including the weightings and bias, are considered as vectors, and the covariance matrix K of all the chromosomes are formed based on the squared exponential kernel function:

$$\mathrm{K}(x, x') = \sigma \cdot \exp\left(-\frac{\| x - x' \|^2}{2l^2}\right) \tag{2}$$

where the x and $x\prime$ are the chromosome vectors. The parameter σ and the characteristic length l are both set as 1. An eigenvector analysis is applied to the K matrix. The super chromosome, called the PCA gene, is obtained as the inner product of the best chromosomes and the eigenvector of the first eigenvalue.

3. The AI-Assisted Simulation Framework and FE Datasets Preparation

In this section, a practical engineering case will be applied to analyze the capability of GA and PCA to generate the initial parameters of the ANN and RNN. However, due to the multiphysical and multiscaled characteristics of the engineering questions, this research obeys a reliable AI-assisted simulation framework [1–3], illustrated in Figure 3, to improve the predicting accuracy of the AI model.

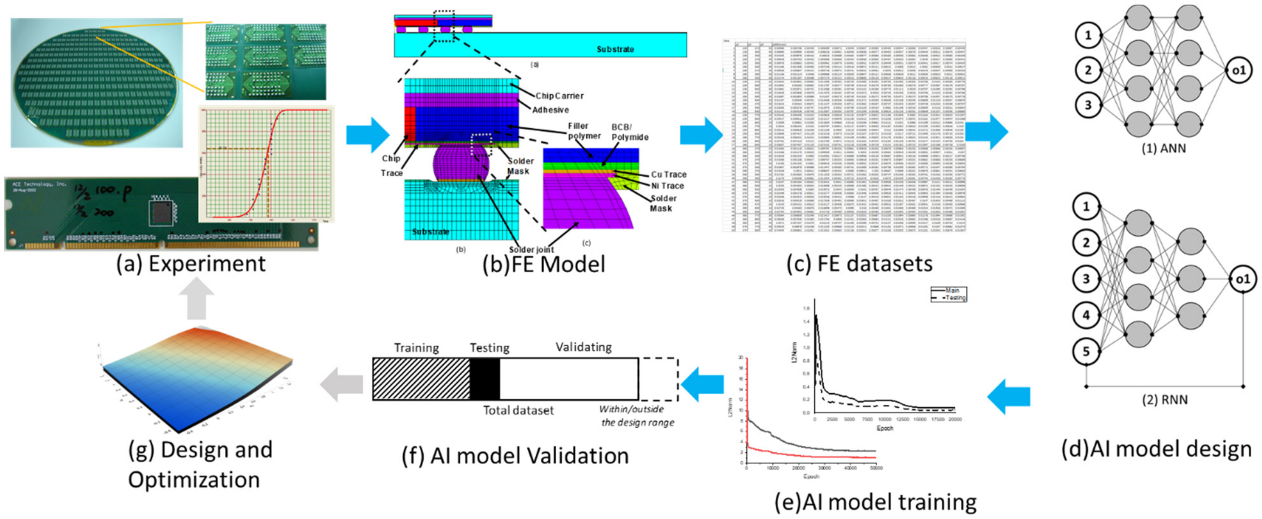

Figure 3. The AI-assisted simulation framework [3,14] (**a**) Experiment; (**b**) FE model; (**c**) FE datasets; (**d**) AI model design; (**e**) AI model training (**f**) AI model validation; (**g**) Design and optimization.

Due to the limited resources, only limited actual samples with very few design parameter combinations are made for the experiments. The FE modeling method is applied to expand the design domain. Based on the experimental results (Figure 3a), a FE model (Figure 3b) can be established and validated. The validated FE model then can be parameterized, and the FE datasets (Figure 3c) can be obtained. However, the specified design parameter combination might induce bad aspect ratios of the elements, which cause the instability of the FE analysis results. The NN model is expected to broaden the design domain. The NN will be carefully designed (Figure 3d) based on the characteristics of the FE dataset and supervised trained (Figure 3e). Moreover, the AI model training's accuracy requirement(s) should be carefully defined based on the FE dataset. The datapairs that have not been included in the training procedure will be applied to validate the NN model. The validated NN model can be used for design and optimization (Figure 3f). When the new experimental result is available (Figure 3g), the whole procedure can be relaunched.

Figure 4 shows the G-WLCSP structure [14], where the IC is placed on a glass substrate with metal traces and solder bumps Figure 4a for redistribution purposes. Figure 4a',b show the device's top view and cross-section view, respectively. Moreover, Figure 4b depicts the key structural components in G-WLCSP, including the glass substrate, the adhesive, the IC, polyimide (PI) for the stress buffer layer, solder mask, and the solder. After the wafer has been diced, individual packaging can be obtained.

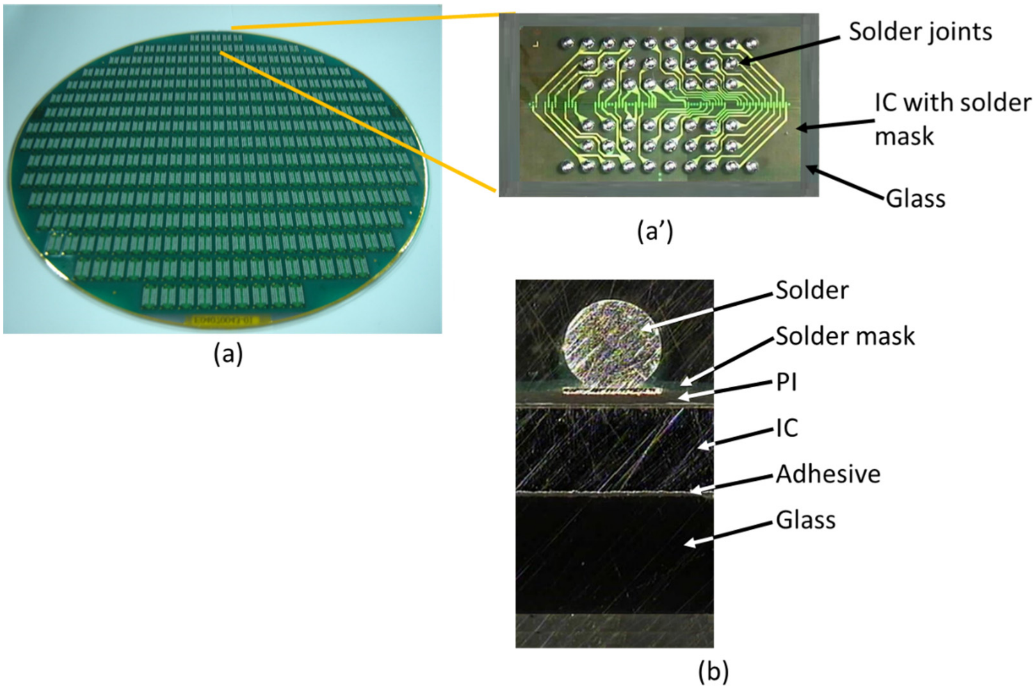

Figure 4. Glass distributed wafer level packaging: (**a**) wafer view with the detail (**a'**) and (**b**) cross-section of a single device [15] (Copyright 2020 EuroSimE).

An actual G-WLCSP structure was made for reliability testing. The sample consisted of a chip with the size of 5.77 × 10.38 × 0.3 mm^3 and a glass thickness of 0.5 mm. The sample was attached to a 1.2 mm-thick test board, as shown in Figure 5a. The cross-section view shows the bonding condition, and no defect has been detected (Figure 5b). A 0.45 mm-diameter 63Sn/37Pb solder ball was applied onto the 0.37 mm die-side pad. The stand-off height for solder joints was reduced to 0.35 mm after reflow.

Thermal cycle testing of this G-WLCSP is performed between −40 °C and 125 °C with a ramp rate of 11 °C/min and a dwelling time of 15 min. Figure 5c shows the Weibull solder fatigue failure experimental result of 21 samples; the 63.2% fatigue cycle number is approximately 1444.

We develop a two-dimensional FE model with a plane strain assumption to estimate the G-WLCSP solder joint risk under the thermal cycle loading by the incremental plastic strain. The initial stress-free reference temperature equals 25 °C. In the finite element model, all materials except the solder joint and the PI are linear, as shown in Table 1. Moreover, the solder joints and PI are treated as temperature-dependent, elastic-plastic materials [15,16], as shown in Figures 6 and 7, respectively. As seen in Liu et al. [16], the solder joint failure risk can be estimated at a certain accuracy level without the time-dependent material properties. Due to the symmetrical condition, one-half of the full-scaled two-dimensional FE model is used, and the analysis result is obtained by the commercial finite element code ANSYS® (version 15, ANSYS, Inc., Canonsburg, PA, USA). The mesh density of the most critical solder is shown in Figure 8.

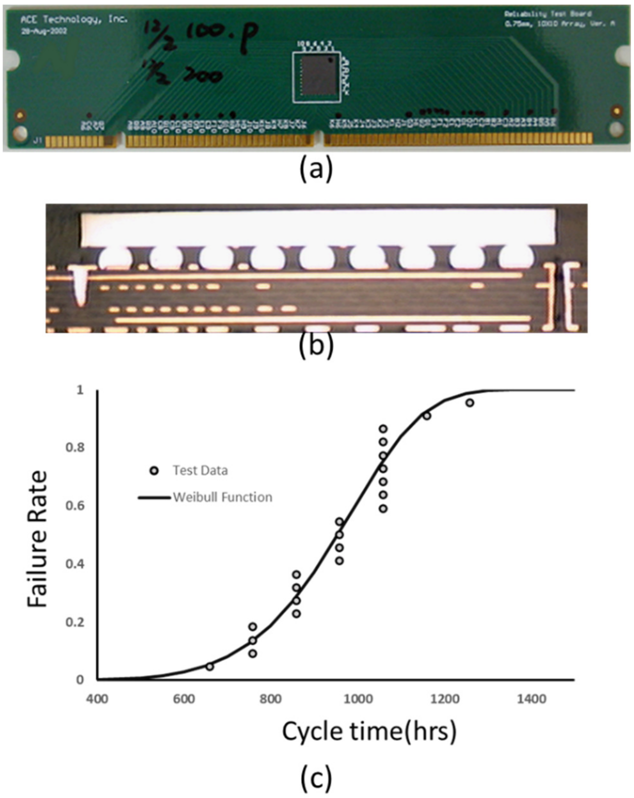

Figure 5. Thermal cycling results: (**a**) the test board, (**b**) the Weibull plot, (**c**) Weibull solder fatigue failure experimental result of 21 samples.

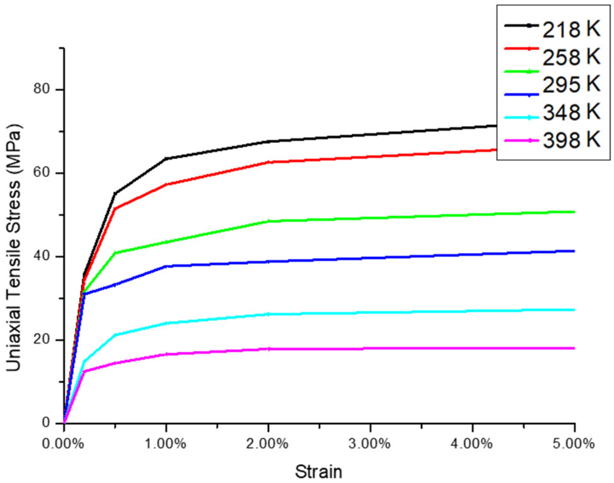

Figure 6. Nonlinear mechanical response of solder joint (63Sn/37Pb) with different temperatures [15] (Copyright 2020 EuroSimE).

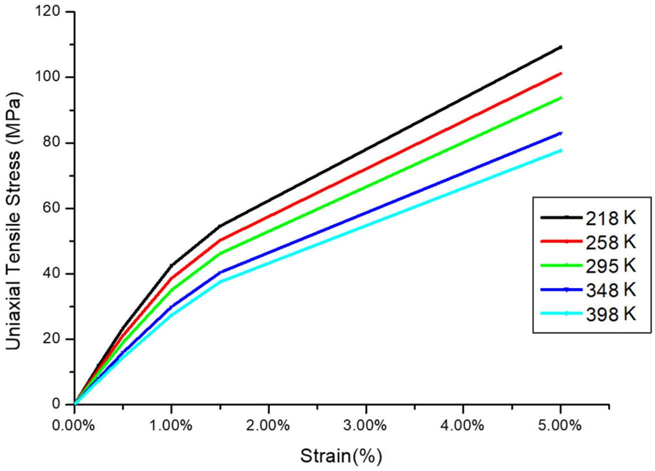

Figure 7. Nonlinear mechanical response of PI with different temperatures [15] (Copyright 2020 EuroSimE).

Table 1. Material properties [14].

	Young's Modules (MPa)	CTE (ppm)	Poission's Ratio
Solder joint	Temperature dependent and nonlinear (Figure 6)	23.90	0.35
Copper	76,000	17.00	0.35
Solder mask	3400	30.00	0.35
FR4 PCB	18,200	16.00	0.19
PI (Stress Buffer Layer)	Temperature dependent and non-linear (Figure 7)	150.00	0.40
BCB	3000	50.00	0.34
Silicon	112,400	2.62	0.28
Adhesive	0.7	300.00	0.45
Glass	63,000	3.25	0.28
Epoxy	80	250.00	0.34

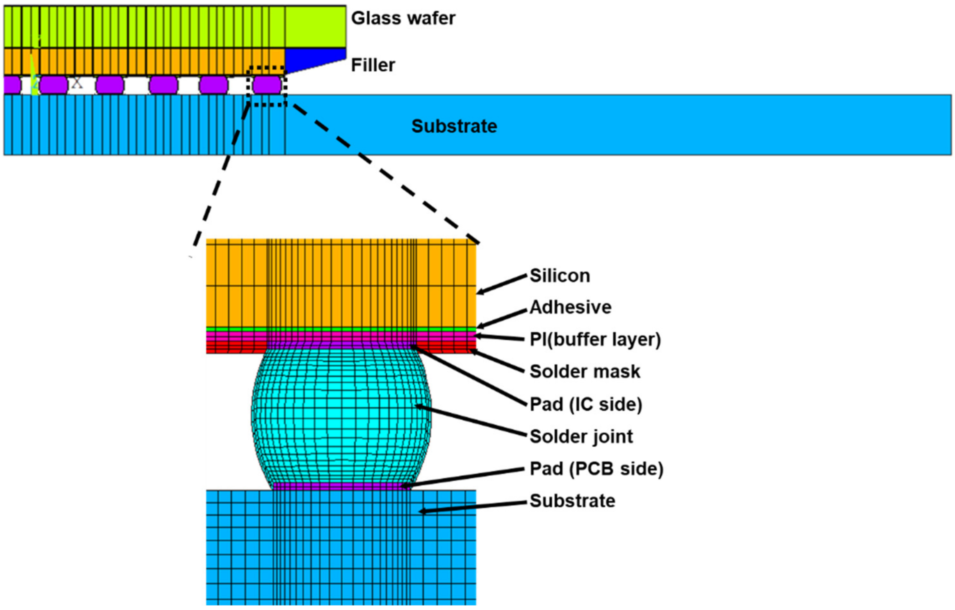

Figure 8. Finite element model for conventional WLCSP and proposed glass WLCSP.

By fine-tuning the mesh density and the solution parameters, the FE model can achieve good agreement with the experimental results by the empirical Coffin–Mason equation [17–19], shown in Table 2. Afterward, the validated FE with the solution parameters can be parametrized.

Table 2. Finite element model validation.

	Experimental Result (Cycles)	Simulation Prediction (Cycles)
Cycles	1007	1444

Key parameters with the levels and noise factors are listed in Table 3. These three key design parameters have been chosen in response to the packaging industry requirements and manufacturing capabilities. A finite element model based on experimental validation is first used to broaden the domain of parameters, and then the neural network model is applied. Each simulation comprises a complete five thermal cyclic loading. To build the FE dataset, 81 parametric finite element models, according to the parameters in Table 3, are executed with a controlled mesh density of the most critical solder joint.

Table 3. Parametric model settings.

Parameter Name	Parameter Alias	Level 1	Level 2	Level 3	Noise Factor Levels
Die thickness	1	0.25 mm	0.375 mm	0.5 mm	±0.015 mm
Glass thickness	2	0.3 mm	0.5 mm	0.8 mm	±0.03 mm
PI thickness	3	0.04 mm	0.025 mm	0.015 mm	±0.005 mm

Figure 9 shows the averaged incremental plasic strain of each loading cycle, where the plastic strain is only induced by plasticity deformation. After the third cycle, the averaged incremental plastic strain incremental becomes stable. From these 81 data points, the average strain increment is 3.01% ($\Delta\epsilon_{avg}$), with a standard deviation of 1.17% (σ_ϵ). The empirical Coffin–Mason equation [17–19] converts to the reliability cycles, as

$$N_f\left(\Delta\varepsilon_p\right) = 0.4405\cdot\left(\Delta\varepsilon_p\right)^{-1.96} \tag{3}$$

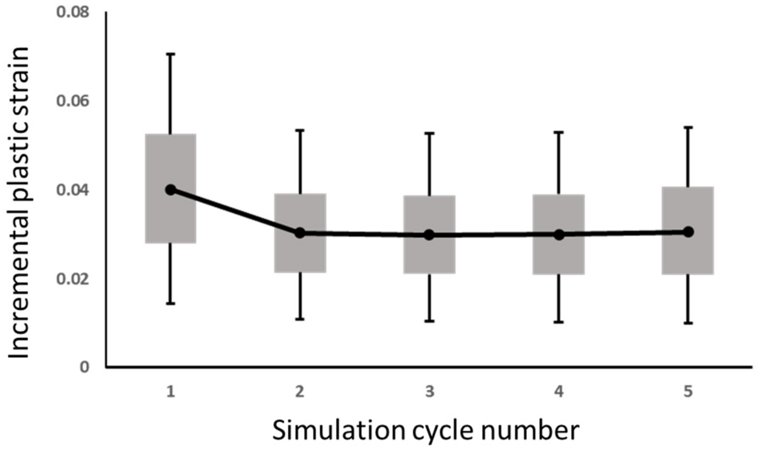

Figure 9. The boxplot of the plastic strain incremental of the 81 data points [15] (Copyright 2020 EuroSimE).

On the other hand, Table 2 indicates that the difference between the experimental and simulation result is 437 cycles. Based on Equation (3), we define a max-min problem:

$$f = \underbrace{\arg}_{\Delta\varepsilon_{accu}\in\mathbb{R}} \left\{\min\left[\max\left(\Delta N_f - \Delta\right)\right]\right\} \tag{4}$$

where ΔN_f is defined as $\left|N_f(\epsilon + \Delta\epsilon_{accu}) - N_f(\epsilon - \Delta\epsilon_{accu})\right|$, $\epsilon = \mathcal{N}\left(\Delta\epsilon_{avg}, \sigma_\epsilon\right)$, and $\mathcal{N}(\cdot,\cdot)$ represents the Gaussian distribution. The $|f|$ in Equation (4) is expected to be zero. By the regression computation, $\Delta\epsilon_{accu} \sim 0.18\%$ is obtained, and it is assigned as the accuracy requirement for machine learning.

4. The Design of the AI Model

Figure 10 schematically illustrates the characteristics of the datapairs. There are three design parameter inputs, including the die, glass, and PI thickness. Since the FE model is under five thermal cycle loadings, denoted as the cycle 1–5 illustrated in Figure 10. At each cyclic thermal loading, there is an equivalent plastic strain ($\Delta\epsilon_{pl}$) with respect to each temperature.

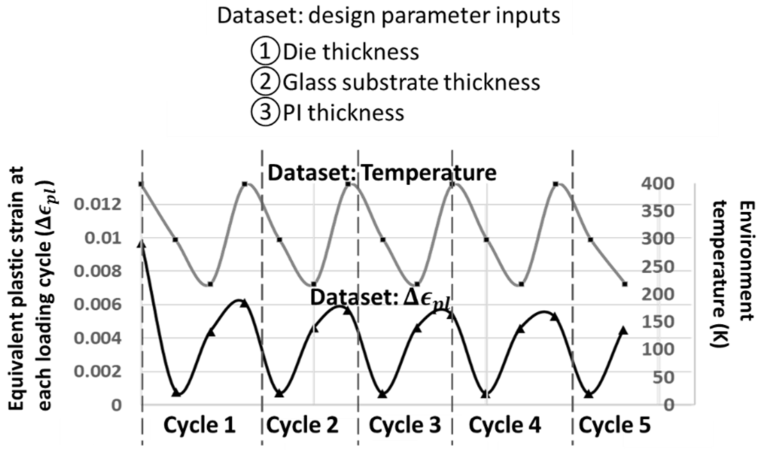

Figure 10. The FE dataset.

Hence, the ANN structure is designed as follows: the three geometric design parameters are considered as the inputs. Referring to the plastic strain incremental of the 81 data points shown in Figure 9, the average equivalent plastic strain increment of the last three loading cycles is selected as the output. The design concept of ANN is to capture the relationship directly from the design parameters to the solder joint fatigue cycle, which is represented by the equivalent plastic strain and converted by the Coffin-Manson equation (Equation (3)).

The ANN structure is designed as "3,4,4,1". There are three inputs and one output, and there are two hidden layers, including the weightings and bias, to capture the feature characteristics of the training datasets. The sigmoid is selected as the activation function because it is stable for the initial parameter studies in the next paragraph. To keep the simplicity during learning, the ANN optimizer is limited to forward computation and backpropagation. The data normalization is applied to the datasets.

Each prediction error is defined as $e = o - t$, where o is the prediction obtained from the ANN output, and t is the ground truth from the FE datasets. The cost function is defined as the Euclidian sum of each prediction error:

$$C = \sqrt{\frac{\sum_{i=1}^{n} e_i}{n}} \tag{5}$$

where e_i is the prediction error of i^{th} prediction and there are total n predictions.

On the other hand, the plastic strain accumulated from the previous cycle will impact the system's mechanical response. Hence, RNN is applied to predict the equivalent plastic strain at each cycle. The equivalent plastic strain (per cycle) is the output. Moreover, there are five inputs, including the three geometrical design parameters, the temperature of the current cycle, and one recurrent parameter. Referring to the top-right schematic drawing in Figure 3, the recurrent parameter of RNN converts the previous output of equivalent plastic strain into the input of the next cycle.

The structure of RNN is set to "5,4,3,1". There are five outputs and one input, and two hidden layers with four and three neurons, respectively. The design concept of the hidden layer is to keep similar numbers of weightings and bias, to compare to the ANN results directly. The sigmoid is selected as the activation function. To maintain the simplicity during learning, the ANN optimizer is limited to forward computation and backpropagation through time. The data normalization is applied to the datasets. One extra post-processing is applied to the ANN. Only the average of the last three outputs is considered to compare to the ANN prediction accuracy directly.

5. AI Model Training with GA Optimized Initial Parameters

In neural network-based AI modeling, the initial parameters are required to launch the machine learning process. This initial parameter is usually chosen randomly in the literature. In this paper, a GA is proposed to achieve the best initial parameter for the AI modeling. The GA chromosome is defined as the combination of genes. Each gene is the combination of the weightings or bias between two layers [12]. For instance, if there are two layers with l_{m-1} and l_m neurons, the gene occupies $l_m \cdot l_{m-1}$ individuals as illustrated in Figure 11. Each individual in the gene is defined as a base-pair (bps). The chromosome is a combination of genes from various layers.

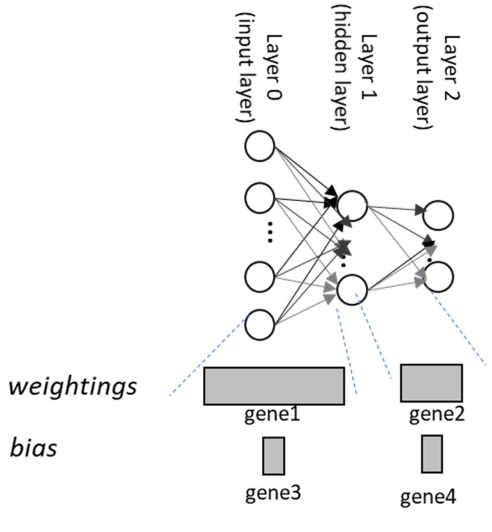

Figure 11. The definition of the chromosomes in GA.

The GA optimization procedure is illustrated in Figure 12. Each GA step starts with the old population of many chromosomes (Ch). The fitness ranking by Equation (1) limits that only m the best chrmosomes can enter the next population. By the crossover and mutation operators, a new population can be generated. The backpropagation of NN learning provides new fitness rankings for the next GA step.

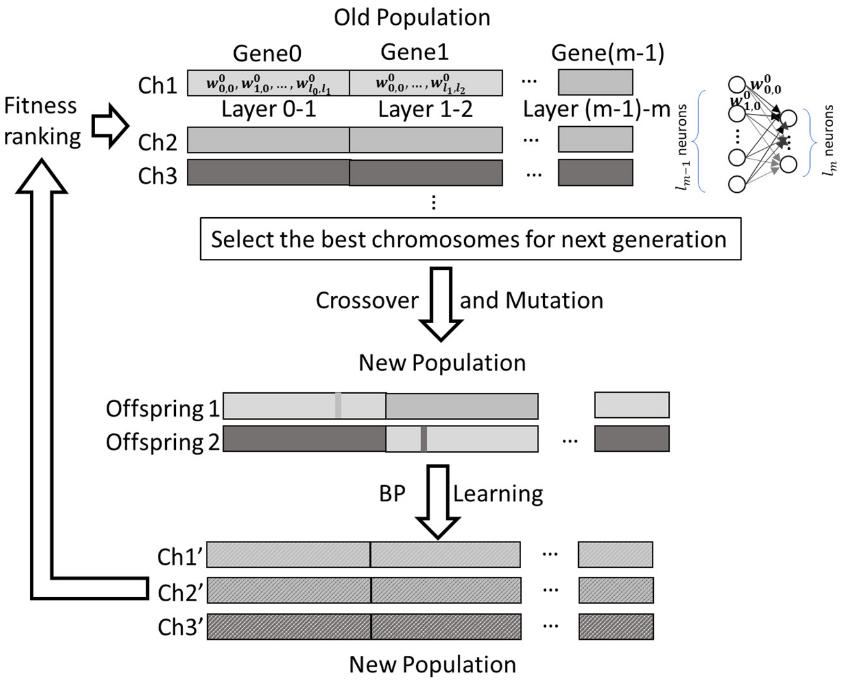

Figure 12. The GA flow chart.

In the AI-learning algorithm, the parameters, including the weightings and bias, are updated each iteration by the backpropagation process. Hence, there is an option to select which parameter sets generate the next GA generation. In this research study, two conditions are considered:

- "back-to-original" condition: only the parameters that initially input to the AI model are applied to generate the next generation.
- "progressing" condition: the parameters that after *n* backpropagation iterations are applied.

5.1. The "Back-to-Original" GA Optimizer

Due to the design of the neural network structure, 2000 initial parameter sets are firstly generated by the random generator, which follows a zero-mean Gaussian distribution of the standard deviation of $\sqrt{2/(l_m \cdot l_{m+1})}$ $(0 \leq m \leq n)$, where n is the total layer number, and l_m is the neuron numbers at m^{th} layer. It also indicates that 2000 initial chromosomes have been generated for each case to initialize the GA optimization procedure. The learning rate of ANN is fixed to 0.3 with the sigmoid activation function, and backpropagation is selected as the learning optimizer.

When the RNN architecture is applied with the "back-to-original" GA optimizer, Figure 13 shows convergence curves among random select chromosomes from the 2000 members (random selection), the best chromosome from the first generation (generation 1 best), and the best one from the whole GA optimization (GA best). A clear contribution of the GA optimization to the convergence speed can be confirmed.

A typical performance of the GA generations under the "back-to-original" optimizer is illustrated in Figure 14, where the lightest grey curve is best of the 2000 initials, and it evolves continuously from the light grey to the darker ones by the GA. The mutation rate for generating the next generation is fixed by 0.001. After the seventh generation, the evolving of the best chromosome stops. The inset of Figure 14 shows the average error norm among the GA generation and generation size of the GA. The average error norms reduce through each generation. The size of GA generation reduces accordingly as the duplicated chromosomes in the generation are removed. Moreover, the standard derivation of average norm varies during the GA optimization, but it dramatically decreases when the GA evolving stops (GES). This is because the optimized chromosome dominates the next GA generation.

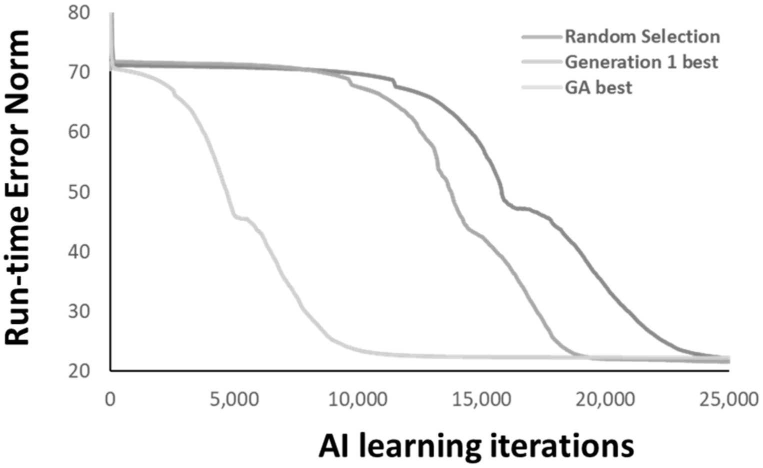

Figure 13. The contribution of GA optimization.

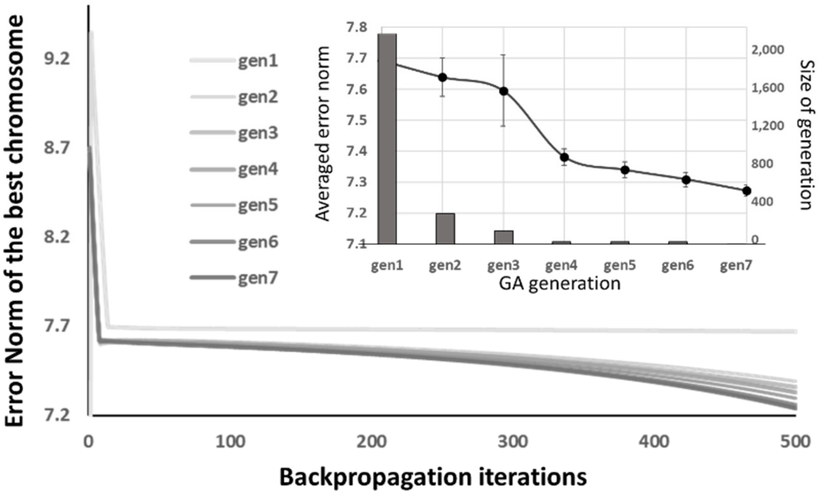

Figure 14. The performance of the GA generations under the “back-to-original” optimizer.

We conducted four GA optimization procedures and listed them in Table 4, where the “Run-time error norm” column indicates the error norm of the best chromosome against the training sets after the 500 cycle training and before the denormalization.

Table 4. The ANN training result under GA “back-to-original” optimizer.

	The Best Chromosomes for the Next Generation	GES	Run-Time Error Norm	Denormalized Difference
Case 1	4	6	7.47	0.001530
Case 2	4	6	7.48	0.001287
Case 3	6	7	7.42	0.001225
Case 4	6	7	7.39	0.001098
PCA gene	–	–	1.72	0.001264

Case 1 and 2 introduce the four best chromosomes to the next generation, and Case 3 and 4 introduce six. Due to that, the initial 2000 members were generated independently by the random number generator for each case in Table 4, a different optimization result is achieved under the same learning parameters. This phenomenon is clearly depicted in column "Run-time error norm" of Table 4.

After the GA optimization procedure, a PCA process is applied based on these four best chromosomes. The covariance matrix K is formed by Equation (2). An eigenvector analysis is applied to the K matrix. Then, The eigenvector of the first eigenvalue is selected and applied to those four best chromosomes. The PCA gene can be obtained by

$$[\boldsymbol{ch}_1,\ \boldsymbol{ch}_2, \boldsymbol{ch}_3, \boldsymbol{ch}_4]\cdot\overrightarrow{\boldsymbol{v}_e} \tag{6}$$

where $\boldsymbol{ch}_1$, $\boldsymbol{ch}_2$, $\boldsymbol{ch}_3$, and $\boldsymbol{ch}_4$ are the best chromosomes from the GA optimization, and $\overrightarrow{\boldsymbol{v}_e}$ is the eigenvector of the first eigenvalue. The convergence curve is shown in Figure 15, where a relatively fast convergence can be achieved. Moreover, from the inset of Figure 15, which extended the convergence curve to 100k iteration, one can identify that these best chromosomes approach similar error norm levels after considerable iteration numbers.

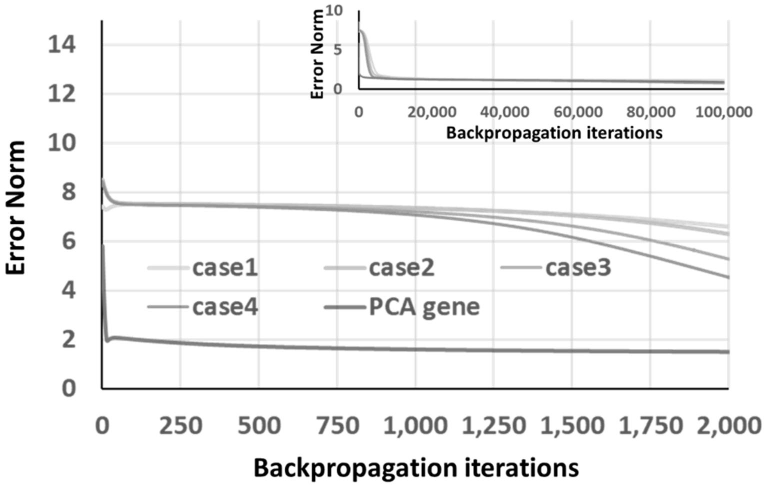

Figure 15. The convergence of 4 GA "back-to-original" optimized chromosomes and PCA gene under ANN architecture.

The "Denormalized difference" column in Table 4 indicates the difference between the best chromosome against the whole 81 datasets after 100,000 training cycles and after the denormalization process. Moreover, no matter the "Run-time error norm", the best chromosomes can consistently achieve the 0.18% requirement. Moreover, when more best chromosomes are put into the next generation, the denormalized difference improves.

Under the similar GA condition of "back-to-original", the RNN architecture has been implemented. Each GA generation is optimized by 1000 backpropagation through time iterations. The rest of the learning parameters remain the same as the ANN case. Table 5 lists the learning results of the four GA optimizations. Note that the run-time error norm is much higher than the ANN ones (listed in Table 4); this is because the definition of the run-time error norm in RNN consists of all recurrent cycles. Figure 10 shows that there are 15 recurrent cycles.

Table 5. The RNN training result under GA "back-to-original" optimizer.

	The Best Chromosomes for the Next Generation	GES	Run-Time Error Norm	Denormalized Difference
Case 1	4	6	69.92	0.002017
Case 2	4	2	70.72	0.001189
Case 3	6	4	69.61	0.001343
Case 4	6	3	70.05	0.001226
PCA gene	–		37.57	0.001090

The PCA analysis procedure is applied to these four GA optimization listed in Table 5. The performance of the PCA gene, obtained by Equation (6) is listed in the last row of Table 5 and Figure 16. Case 1 in Table 5 does not satisfy the accuracy requirement, but case 2 does. Both cases 1 and 2 use the same GA parameters but different initial GA generation. Hence, the GA optimization results are influenced by the initial GA generation.

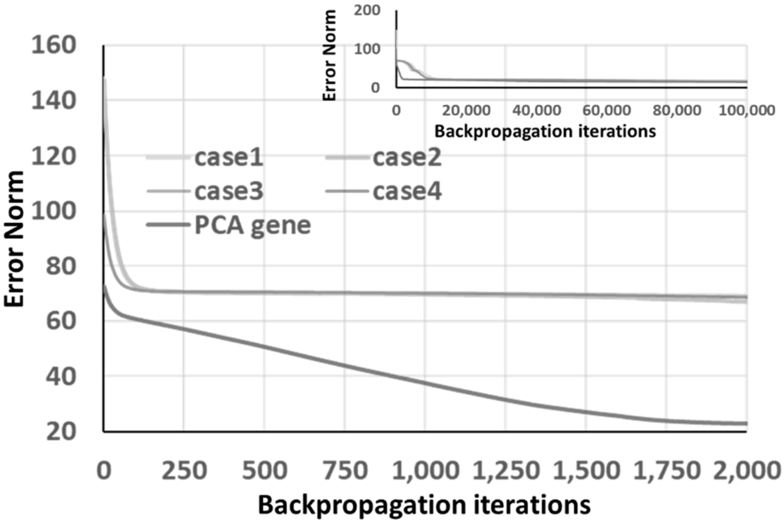

Figure 16. The convergence of 4 GA "back-to-original" optimized chromosomes and PCA gene under RNN architecture.

On the other hand, one can identify the fast convergence capability of the PCA gene, shown in Figure 16. To decrease the impact of the initial GA generation, it is recommended that one should execute multiple GP optimization processes and then extract these best chromosomes to the PCA gene. Moreover, although RNN provides more information than ANN, both give similar final prediction accuracy, as listed in the last row of Tables 4 and 5.

5.2. The "Progressing" GA Optimizer

The converging behavior of the "progressing" GA optimization demonstrates much difference from the "back-to-original" one. As illustrated in Figure 17, which is the performance of the GA generations under the "progressing" optimizer under an ANN architecture. It clearly depicts the evolving of the chromosomes moving along the backpropagation process. The dashed line of Figure 17 is the convergence curve of the randomly selected 2000 initial GA generation. Comparing the dashes curve to the solid ones, one can identify the continuous optimizing characteristics of the "progressing" GA. The inset of Figure 17 shows the average error norm and size of the generation with respect to the GA generation. Although a continuous convergence of the average error norm can be found, the size of the generation is larger than the "back-to-original" one. This is because the backpropagation algorithm provides various gene combinations with few duplications for the next generation, which might increase the optimization capability of GA with considerable computation resources.

The complete training results are listed in Table 6. All of the denormalized differences satisfy the accuracy requirement of 0.18%. Among all four cases, case 3 performs worse than the other three in terms of the denormalized difference, but it performs well in the run-time error norm. This is because the run-time error norm only reports the learning results of the chromosomes after 500 iterations, but the denormalized difference reflects the 100k learning. This phenomenon depicts the dilemma of selecting how many backpropagation iterations for each GA generation member.

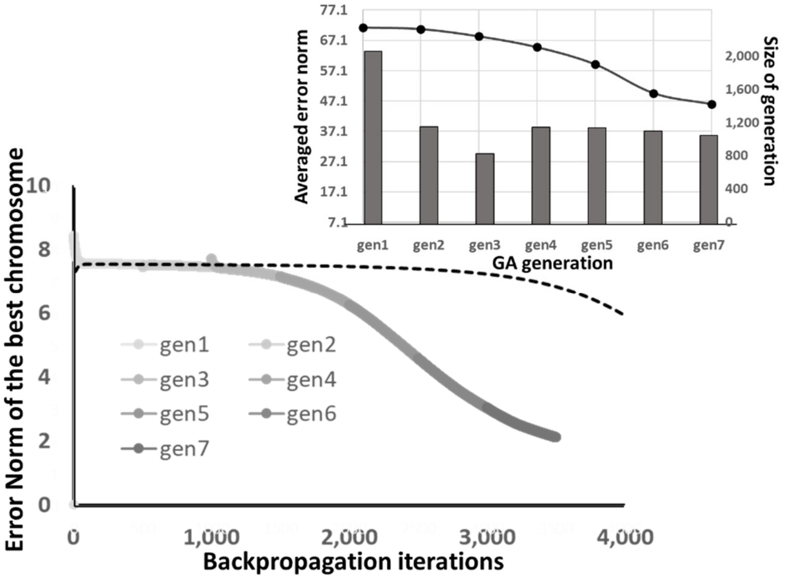

Figure 17. The performance of the GA generations under the "progressing" optimizer.

Table 6. The ANN training result under GA "progressing" optimizer.

	The Best Chromosomes for the Next Generation	GES	Run-Time Error Norm	Denormalized Difference
Case 1	4	6	6.87	0.001058
Case 2	4	4	6.58	0.001037
Case 3	6	6	4.93	0.001573
Case 4	6	8	1.70	0.001175
PCA gene	–	–	1.87	0.001098

Figure 18 shows the convergence of the four cases and the PCA gene obtained by the same method mentioned above. Although case 4 reveals a low run-time error at the beginning of the learning process, the PCA gene still performs well after approximately 750 learning cycles.

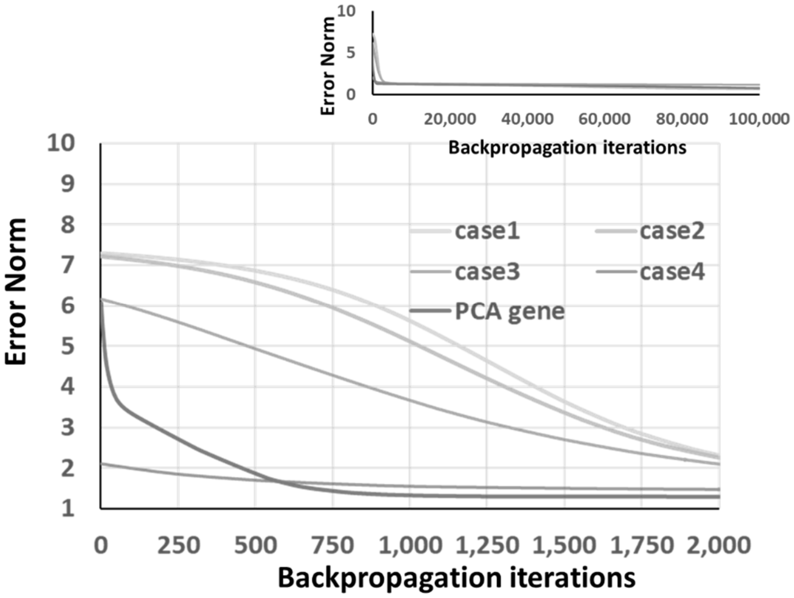

Figure 18. The convergence of 4 GA "progressing" optimized chromosomes and PCA gene under ANN architecture.

5.3. The Impact of the Initial GA Generation

The learning results of the RNN architecture with the “progressing” GA optimizer are shown in Table 7 and Figure 19. Although the overall characteristics, including the convergence curves, the denormalized difference, PCA gene, etc., behave similarly to the previous cases, the overall performance is not good enough, compared to the same RNN architecture with the “back-to-original” GA optimizer, shown in Table 5 and Figure 16.

Table 7. The RNN training result under GA “progressing” optimizer.

	The Best Chromosomes for the Next Generation	GES	Run-Time Error Norm	Denormalized Difference
Case 1	4	8	42.29	0.003281
Case 2	4	6	47.31	0.001663
Case 3	6	6	45.31	0.001592
Case 4	6	7	46.03	0.001254
PCA gene	–	–	26.78	0.001493

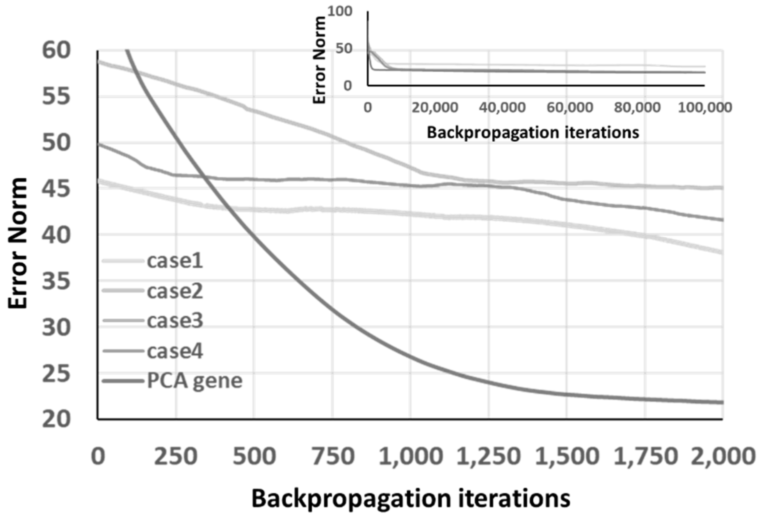

Figure 19. The convergence of 4 GA “progressing” optimized chromosomes and PCA gene under RNN architecture.

In order to investigate the impact of the initial GA generation, the ones belonging to Table 5 are applied to the “progressing” GA optimizer, and the learning results are listed in Table 8 and Figure 20. The PCA gene shows fast convergence capability, as illustrated in Figure 20. Comparing Tables 7 and 8, a clear difference can be identified under the “progressing” GA optimizer with the same GA optimization parameters in terms of run-time error norm and denormalized difference. The influence of the initial GA generation is proven to be significant. Therefore, the recommendation of using the independent initial GA generation for all cases is made.

Table 8. The RNN training result under GA “progressing” optimizer by the same initial 2000 GA generation in Table 5.

	The Best Chromosomes for the Next Generation	GES	Run-Time Error Norm	Denormalized Difference
Case 1	4	4	22.70	0.001406
Case 2	4	9	21.46	0.001022
Case 3	6	9	21.78	0.001263
Case 4	6	6	22.00	0.001204
PCA gene	–	–	20.07	0.001110

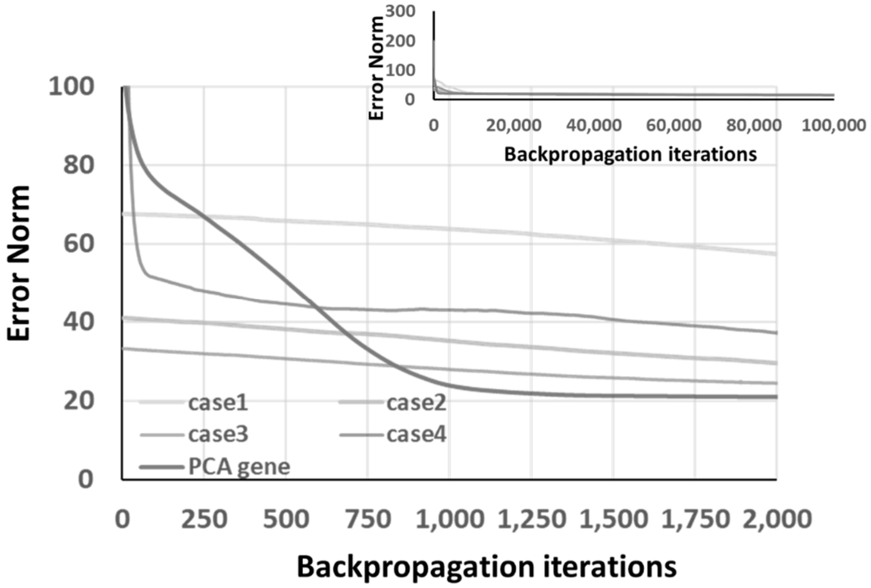

Figure 20. The convergence of 4 GA "progressing" optimized chromosomes and PCA gene under RNN architecture by the same initial 2000 GA generation in Table 5.

6. Conclusions

In this research, the WLCSP solder joint reliability risk is modeling by an AI model, following the AI-assisted simulation framework. ANN and RNN architectures are conducted to investigate their capability of abstracting the time-dependent solder joint fatigue knowledge from the dataset. The GA optimization is applied to decrease the influence of the initial guessings, including the weightings and bias of the neural network architectures. Due to the continuous learning characteristics of the backpropagation, the "back-to-original" and "progressing" GA optimizers are developed.

Both ANN and RNN architectures, with two hidden layers, are conducted with similar neural network structures. Two GA optimizers are applied to both ANN and RNN architectures with four and six best chromosomes to the next generation. Each GA optimization case starts with an independent 2000 initial GA generation, and each component of the chromosomes follows a zero-mean Gaussian distribution. Moreover, a PCA is applied to the GA optimization results to obtain the PCA gene. PCA gene shows high-speed convergence capability in all cases.

The investigation of the GA optimization shows that increasing the number of the best chromosomes to the next generation and choosing the "progressing" GA optimizer improve the GA optimization results. However, both increase a significant computation resource to conduct. The influence of the initial GA generation is proven to be significant. Therefore, using the independent initial GA generation for all cases and using the PCA gene obtained by several GA optimization processes are recommended.

Because ANN and RNN learnings are more robust due to the GA, these neural networks are suitable for generating response surfaces, as seen in Figure 3g. The predictability of the neural network model enables the exploration of the domain that is outside the training domain (the FEM domain) at a certain range due to the contribution of the nonlinear activation functions [3]. Moreover, due to the continuity of the neural network model, these models are feasible for the optimization procedure.

No matter ANN nor RNN architecture, after 100 k learning iterations, all the AI learning results satisfy the accuracy requirement of 0.18% when the PCA gene is applied as the initial parameter. Moreover, there is no clear evidence that RNN is statistically better than ANN in the WLCSP solder joint reliability risk estimation if the PCA gene is applied. Although RNN provides more information than ANN, RNN is influenced by the noise in the dataset during the learning, which limited the RNN to perform much better than

ANN. However, RNN learning requires more computation resources than ANN because of the backpropagation process under a similar neural network structure. Hence, a stable optimization with a broad design domain can be realized by an ANN model with PCA gene with a faster training speed than RNN, even though solder fatigue is a time-dependent mechanical behavior.

Author Contributions: Conceptualization, methodology, software, formal analysis, investigation, data curation, writing—original draft preparation, C.Y.; Problem identification, concept initialization, writing—review and editing, discussion during research, supervision, X.F. and G.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research is partially supported by the "Developing Smart Automation Mechatronics System and Internet of Things for Shoe Manufacturing" project of Feng Chia University, sponsored by the Ministry of Science and Technology, under the grand no. MOST109-2218-E-035-002, and the "Research of the industrial internet of things for the continuous manufacturing with heterogeneous machines, the production status prediction model and the smart decision-making algorithm" project of Feng Chia University, sponsored by the Ministry of Science and Technology, under the grand no. MOST 109-2221-E-035-008-MY2.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this research study are available in this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chou, P.H.; Chiang, K.N.; Liang, S.Y. Reliability Assessment of Wafer Level Package using Artificial Neural Network Regression Model. *J. Mech.* **2019**, *35*, 829–837. [CrossRef]
2. Hsiao, H.Y.; Chiang, K.N. AI-assisted reliability life prediction model for wafer-level packaging using the random forest method. *J. Mech.* **2020**, *37*, 28–36. [CrossRef]
3. Yuan, C.; LEE, C.-C. Solder Joint Reliability Modeling by Sequential Artificial Neural Network for Glass Wafer Level Chip Scale Package. *IEEE Access* **2020**, *8*, 143494–143501. [CrossRef]
4. Yuan, C.C.A.; Fan, J.; Fan, X. Deep machine learning of the spectral power distribution of the LED system with multiple degradation mechanisms. *J. Mech.* **2021**, *37*, 172–183. [CrossRef]
5. Meszmer, P.; Majd, M.; Prisacaru, A.; Gromala, P.J.; Wunderle, B. Neural networks for enhanced stress prognostics for encapsulated electronic packages-A comparison. *Microelectron. Reliab.* **2021**, *123*, 114181. [CrossRef]
6. Selvanayagam, C.; Duong, P.L.T.; Raghavan, N. AI-Assisted Package Design for Improved Warpage Control of Ultra-Thin Packages. In Proceedings of the 2020 21st International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE), Cracow, Poland, 5–8 July 2020; pp. 1–7.
7. Tabaza, T.A.; Tabaza, O.; Barrett, J.; Alsakarneh, A. Hysteresis modeling of impact dynamics using artificial neural network. *J. Mech.* **2021**, *37*, 333–338. [CrossRef]
8. Mitchell, T.M. *Machine Learning*, 1st ed.; McGraw-Hill: Boston, MA, USA, 1997.
9. White, D.; Ligomenides, P. GANNet: A genetic algorithm for optimizing topology and weights in neural network design. In Proceedings of the New Trends in Neural Computation, Sitges, Spin, 9–11 June 1993; pp. 322–327.
10. Juang, C.F. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Trans. Syst. Man Cybern. Part B (Cybernetics)* **2004**, *34*, 997–1006. [CrossRef] [PubMed]
11. Ding, S.; Su, C.; Yu, J. An optimizing BP neural network algorithm based on genetic algorithm. *Artif. Intell. Rev.* **2011**, *36*, 153–162. [CrossRef]
12. Ahmadizar, F.; Soltanian, K.; AkhlaghianTab, F.; Tsoulos, I. Artificial neural network development by means of a novel combination of grammatical evolution and genetic algorithm. *Eng. Appl. Artif. Intell.* **2015**, *39*, 1–13. [CrossRef]
13. Arabasadi, Z.; Alizadehsani, R.; Roshanzamir, M.; Moosaei, H.; Yarifard, A.A. Computer aided decision making for heart disease detection using hybrid neural network-Genetic algorithm. *Comput. Methods Programs Biomed.* **2017**, *141*, 19–26. [CrossRef] [PubMed]
14. Yuan, C.-A.; Han, C.N.; Yew, M.-C.; Chou, C.-Y.; Chiang, K.-N. Design, analysis, and development of novel three-dimensional stacking WLCSP. *IEEE Trans. Adv. Packag.* **2005**, *28*, 387–396. [CrossRef]
15. Yuan, C.; Lee, C. Solder joint reliability risk estimation by AI modeling. In Proceedings of the 2020 21st International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE), Cracow, Poland, 5–8 July 2020.

16. Liu, C.-M.; Lee, C.-C.; Chiang, K.-N. Enhancing the Reliability of Wafer Level Packaging by Using Solder Joints Layout Design. *IEEE Trans. Compon. Packag. Technol.* **2006**, *29*, 877–885. [CrossRef]
17. Coffin, L.F., Jr. A Study of the Effects of Cyclic Thermal Stresses on a Ductile Metal. *Trans. ASME* **1954**, *76*, 931–950.
18. Manson, S.S. *Thermal Stress and Low Cycle Fatigue*; McGraw-Hill Book Co., Inc.: New York, NY, USA, 1966.
19. Solomon, H.D. Fatigue of 60/40 Solder. *IEEE Trans. Compon. Hybrids Manuf. Technol.* **1986**, *9*, 91–104. [CrossRef]