 https://doi.org/10.1109/TCAD.2022.3216549

Benchmarking of Machine Learning Methods for Multiscale Thermal Simulation of Integrated Circuits

David Coenen[1,2], Herman Oprins[2], Robin Degraeve[2], Ingrid De Wolf[1,2]
[1]Dept. Materials Eng. KU Leuven, Leuven, Belgium, [2]imec, Kapeldreef 75, Leuven, Belgium
David.Coenen@imec.be

***Abstract*—Multiscale thermal analysis in integrated systems is required for capturing both device-level and circuit-level dynamics. Traditional analysis with finite element (FE) models can be accelerated by using machine learning (ML) methods. In this paper a performance benchmarking between three ML methods for thermal simulation is carried out: Artificial Neural Networks (ANNs), Proper Orthogonal Decomposition with Radial Basis Functions (POD-RBF) and finally POD-RBF-ANN is used as a hybrid ML method. The (dis)advantages of the different methods are demonstrated for the thermal simulation of a multiscale photonic chip. The ML models are trained with FE data for both linear and non-linear dynamics and are tested for inter- and extrapolation prediction accuracy. A computational speed increase with factor >7500 compared to FE is obtained. Furthermore, ANNs prove to be the best suited for the simulation of non-linear dynamics. POD-RBF is the best method for minimizing training time and combining the best of both methods in POD-RBF-ANN creates a ML model with short training phase and highly accurate predictions.**

***Index Terms*—Machine Learning, Thermal Modelling, Silicon Photonics**

I. Introduction

THE increasing complexity and size scaling in today's nanoelectronics and -photonics comes with the need for multiscale analysis from nanometer to centimeter scales [1] and temporal analysis from nanosecond to second scales. Widely used modelling tools such as finite element models for thermal simulation [2] are lacking in this regard as the number of required elements and hence the computational time rapidly increases for capturing details at the smallest scales [3]. In order to solve this problem, there have been many efforts in recent studies to make the simulation process more efficient [4]. Recent fast compact thermal modelling methodologies for the temperature distribution prediction in 2D and 3D chip packages include the Hotspot tool [5], [6] and the Power Blurring method [7], [8]. HotSpot is a compact model based on a discretized thermal resistor-capacitor network, typically used for fast thermal simulation of different CPU architectures, including the package and cooling boundary conditions. Thermal RC-networks are lumped parameter representation of more complex systems and leverage already existing circuit simulators. There are two main methods of extracting the RC equivalent circuit: firstly, there is spatial discretization and calculation of the resistances based on thermal conduction in the structure (e.g. HotSpot) [9], [10], and secondly the resistances of a Foster equivalent network can be calculated by analyzing the time constant spectrum of the system [12], [13]. Power Blurring is a semi- analytical method based on the Green's function of the heat conduction equation. The method uses convolution between the spatial and temporal [14] temperature responses (thermal mask) to a unit cell power dissipation and the distributed power map to calculate the temperature distribution. This method has been applied to 2D and 3D chip packages, power electronics [15], and additions have been proposed to include the impact of 3D interconnect structures and the impact of the chip package [16]. The general idea of all aforementioned compact models is to make a reduced-order representation where the total number of degrees of freedom (DOFs) is reduced to a minimum [11], which in turn speeds up the simulation process. Machine learning (ML) methods for regression are a popular choice as they have been proven to be very efficient [19]. Artificial Neural Networks (ANNs) are already used in thermal simulations, but are considered to be black box methods [20]–[24]. The neurons in the hidden layers and their corresponding attributes such as weights and biases are obtained through the process of back-propagation and fitting to a training dataset. Consequently, these hidden layers do not necessarily represent any physical characteristics of the system. This is an important drawback for the interpretability of the model. On the other side of the spectrum, there are the white box ML methods [25] that contain physical information or governing equations about the system. These typically contain much less trainable parameters and are easier to interpret, making them closely related to traditional statistics and not only ML. Finally, there exist hybrid ML methods, that combine artificial neural networks with more classical statistical methods [21], [24], [28], [29]. What is still missing however, is a performance benchmarking of those ML methods and training dataset design rules specifically for multiscale thermal simulation. Circuit design tools could be extended with compact ML models in order to assess thermal performance early on in the design stage.

Proper Orthogonal Decomposition (POD) combined with Radial Basis Functions (RBF) is one example of a ML method that uses classical statistical tools. In this approach the fundamental behaviour or patterns (POD modes) are retrieved from the training dataset as a first step [25].

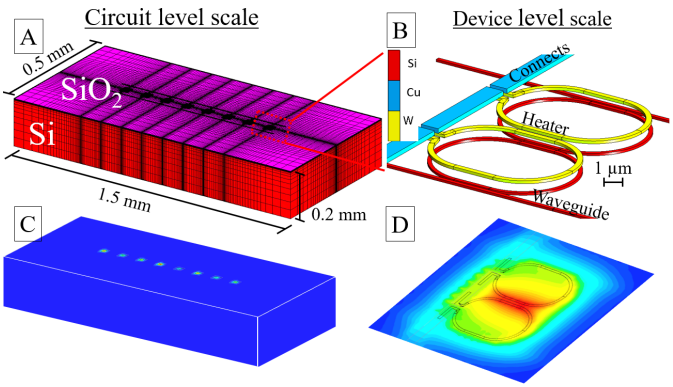

Fig. 1. (A) Finite element model of SiPho die with DEMUX filter array. (B) Zoom-in on a single device, (C,D) temperature contour plots of the full circuit and device respectively.

TABLE I
OVERVIEW OF DIFFERENT ML METHODS. REFERENCE WITH ASTERIX (*) IS WITH A SIMILAR WHITE BOX APPROACH POD-GP (GALERKIN PROJECTION)

	ML methods for multiscale simulation		
	ANN	**POD-RBF**	**POD-RBF-ANN**
Type	Black box	Black box	Black box
Ref.	[20], [22]	[3]* [25]* [37]	[21], [24], [28], [29]

Subsequently combining the POD modes allows for the prediction of new data points. It is a black box method in the sense that no physical information or governing equations are used during the model training. However, it is more interpretable compared to an ANN because the POD modes represent fundamental patterns within the training data, which can be useful information. As a classical black box method, regular feedforward ANNs will be used. Lastly, combining the benefits of the aforementioned methods, a hybrid ML method is also investigated: POD-RBF-ANN employs the ANN for predicting the POD mode coefficients. All previously mentioned ML methods have already been studied extensively, and this work focuses on benchmarking their performance for a specific application (multiscale thermal simulation). As a case study for a multiscale thermal modelling analysis, the thermal behaviour of a photonic chip is used [26]. At nm-µm scale it contains ring-based wavelength filters (channels) of which the temperature distribution with highly localized peak temperature and strong temperature gradient is of interest [27] (Fig.1, right). The full chip itself is at mm scale, and at this scale mainly the thermal coupling between the different channels is of concern. In Section V.A the ML methods will be applied for the thermal simulation of a mobile processor package in order to show that the presented methods are not exclusive for photonic ICs. Lastly, the performance of the proposed ML methods will be compared to the readily available compact modelling tools HotSpot [5] and Power Blurring [7].

II. METHODOLOGY

Machine learning models always require a training dataset to start from. For thermal simulation this can either be generated through simulation by non-ML models, or from experiments. Here we focus on the training data generation with conventional, non-ML finite element (FE) models. The high-level overview of information flow is depicted in Fig.2: the FE simulation result is the input for creating a training dataset. This is fed into the ML layer, where three different methods will be investigated. Once these models are trained, they can be used for predicting the 3-D temperature distribution in the photonic chip in a fast manner, making the use of FE simulation obsolete (except for creating training datasets).

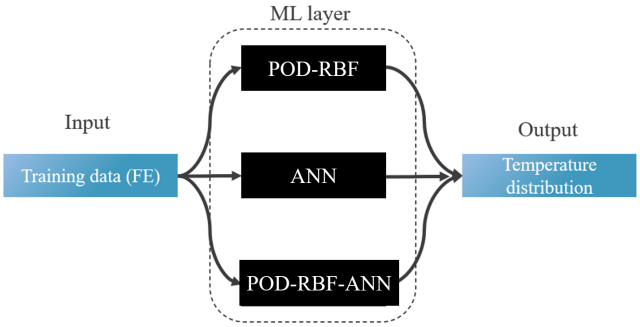

Fig. 2. Flowchart of methodology used in this work, FE = finite element, ML = machine learning.

A. Finite element model

The first case study used in this work is a silicon (Si) photonics die, containing a photonic integrated circuit (PIC) that consists of an 8-channel wavelength filter. Each channel has a cascaded ring resonator for wavelength filtering purposes and is equipped with an integrated tungsten (W) heater for thermal tuning [32], the heater is shown in Fig.1 (b). This is a prime candidate for testing multiscale thermal models, as both the temperature close to the channels and in between is of interest, as well as the thermal coupling between the channels. The training set for the ML models is generated from a thermal finite element simulation of the photonic die using the finite element tool MSC Marc [33]. The constructed FE model for the photonic die test case with 8 channels, shown in Fig.1, consists of 1.5 M nodes . Only heat conduction in the die itself is simulated, heat transfer on either the top or bottom face of the die is modelled with an equivalent heat transfer coefficient. The heat source in the simulation consists of Joule heat generated in the W-heaters in each channel. The value of these boundary conditions will be varied (see Section III). The sidewalls are assumed to be adiabatic. Finally, details on the material properties used in the model can be found in [34].

B. POD-RBF

The first ML method that is discussed, is POD-RBF. This is a method that is often used in the context of model order reduction. The motivation for this is that finite element models consisting of millions of nodes can be replaced with a model of much lower order. For thermal simulations, the shape of the final solution is typically determined by boundary conditions

and heat sources, which opens up the opportunity to describe the system with less degrees of freedom than the original full size computational mesh. The concept of this approach is that the field under investigation (e.g. temperature) is decomposed in fundamental modes. These mode can then in turn be used in a (weighted) superposition to reconstruct the solution: [36], [37]:

$$T(x,y,z,t) = T_0(x,y,z) + \sum_{i=1}^{m} b_i(t)\phi_i(x,y,z), \qquad (1)$$

where T_0 is the average temperature, b_i are the POD coefficients and ϕ_i are the POD modes. The coefficients b_i are not exclusively time-dependent, as indicated in Eq.1, they can depend on a range of input parameters. In order to solve Eq.1, the POD modes need to be calculated, which is achieved using the singular value decomposition (SVD) and their coefficients are obtained with RBF interpolation [37]. The POD modes are orthonormal basis functions. They can be obtained by finding a function Ψ which maximizes the inner product with the temperature field in the training data. The function which accomplishes this, will be the POD modes ϕ [30], [31]:

$$\max_{\Psi} = \left\langle \frac{(\int_\Omega T(\vec{r},t)\Psi d\Omega)^2}{\int_\Omega \Psi^2 d\Omega} \right\rangle = \left\langle \frac{(\int_\Omega T(\vec{r},t)\phi d\Omega)^2}{\int_\Omega \phi^2 d\Omega} \right\rangle \qquad (2)$$

Here Ω represents the physical domain over which is integrated, $\vec{r}$ the spatial coordinates and the brackets $\langle\rangle$ represent the average value. We want to find a basis Ψ which is on average most similar to the original dataset. A necessary condition for the maximisation to hold, is that ϕ is an eigenfunction of the two-point correlation tensor [30]:

$$\int_{\vec{r'}} \mathbf{R}(\vec{r},\vec{r'}) \cdot \vec{\phi}(\vec{r'}) d\vec{r'} = \lambda \vec{\phi}(\vec{r}) \qquad (3)$$

The maximisation in Eq.2 leads to an eigenvalue problem in Eq.3 with eigenvalues λ and eigenvectors $\vec{\phi}$ (POD modes) which can be solved by performing the singular value decomposition on the snapshot matrix. Applying the SVD to the input matrix (snapshot matrix) results in a factorization [37]:

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \qquad (4)$$

Where $\mathbf{M}$ is the snapshot matrix, $\mathbf{U}$ is the left-singular matrix, $\mathbf{\Sigma}$ is a diagonal matrix containing the singular values and $\mathbf{V}$ the right-singular matrix. The singular values are ranked in descending order $\sigma_1 > \sigma_2$ which allows to rank the extracted POD modes in order of importance. The shape of the matrices in Eq.4 is shown schematically in Fig.3. The snapshot matrix $\mathbf{M}$ has n column vectors representing the different training data points, so each column is one full solution generated by finite element simulation. The snapshot matrix also has m rows, which contain the evolution of one specific node's temperature over the whole set of training data points. From the matrix $\mathbf{U}$ the POD modes are extracted and because they are ranked in order of importance, a truncation is possible which eliminates the noise or modes which do not contain any new information. This is the motivation for categorizing this method as model order reduction: the solution is obtained using a truncated set of basis functions. One objective way of determining the cut-off point for truncation is by calculating

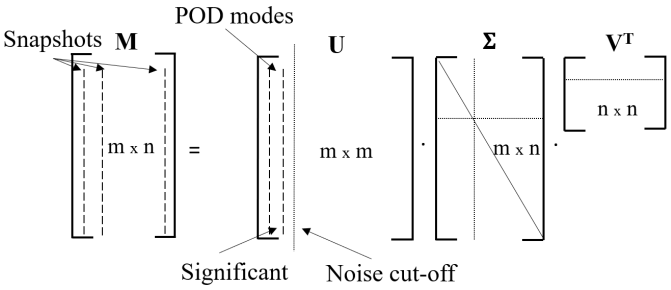

Fig. 3. Snapshot matrix decomposition with singular value decomposition (SVD)

the cumulative correlation energy E_c, also called the relative information content (RIC):

$$E_c = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i} > 0.999 \qquad (5)$$

The eigenvalues λ_i are related to the singular values σ_i in matrix $\mathbf{\Sigma}$ as $\lambda_i = \sigma_i^2$. If E_c is at least 99.9%, then the first m POD modes contain sufficient information. The next step is determining the POD coefficients from Eq.1. One way of doing this is, is by Galerkin projection [3], [25], [36]. This requires solving a system of coupled ordinary differential equations, formed by the heat equation projected on the orthonormal basis. An alternative solution is by using radial basis functions as an interpolation method [37], [38]. RBF is used to reconstruct a function of which a limited number of points are known. As an example, RBF is applied to 1-D data: $s(x)$ is the continuous RBF interpolation function of data known in discrete points $f(x_j),\ j = 1, 2..n$ [38]:

$$s(x) = \sum_{i=1}^{n} a_i \psi(||x - x_i||) = f_j,\ j = 1, 2...n, \qquad (6)$$

where ψ is an RBF function a_i are the RBF weights. The weights depend on the Euclidian distance between x and x_i. The RBF function ψ can take many forms and in general is a function that is maximal for distance zero $||x - x_i|| = 0$ and decays for larger distances. The inverse multiquadratic function is used here:

$$\psi = 1/\sqrt{1 + (\epsilon r)^2}\ with\ r = ||x - x_i|| \qquad (7)$$

ϵ is a tuning parameter that determines the smoothness of the interpolation. The RBF weights a_i are calculated by setting the interpolation function $s(x)$ equal to the values in the known data points, which is followed by matrix inversion in order to solve for a_i:

$$\begin{bmatrix} \psi(r_{1,1}) & \psi(r_{2,1}) & \cdots & \psi(r_{n,1}) \\ \psi(r_{1,2}) & \psi(r_{2,2}) & \cdots & \psi(r_{n,2}) \\ \vdots & \vdots & & \vdots \\ \psi(r_{1,n}) & \psi(r_{2,n}) & \cdots & \psi(r_{n,n}) \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \qquad (8)$$

POD-RBF interpolation is done in similar fashion as this 1-D example: the solution is obtained by finding the best combination of POD modes that fits the new set of input parameters. The distance $||x - x_i||$ can be interpreted as the difference between the new input parameter and the parameter

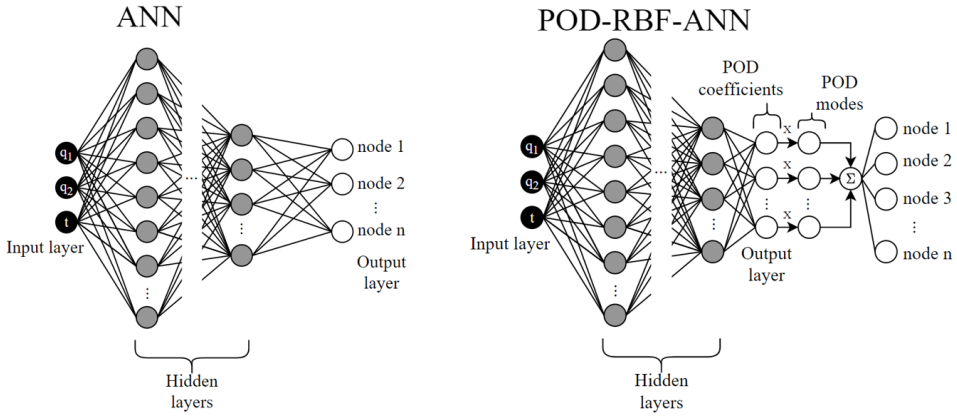

Fig. 4. Schematic representation of ANN (left) and POD-RBF-ANN (right).

for which the solution is known $||p - p_i||$. The practical implementation of the POD-RBF ML model is with the open-source Python package EZyRB [35].

C. ANN

Artificial neural networks exist in many shapes and forms. The focus here lies on fully-connected feedforward ANNs implemented in Python using the Keras [39] and TensorFlow [40] libraries. In Fig.4 (left) the general network architecture is shown. The input layer consists of a number of neurons equal to the amount of input parameters chosen for the FE training dataset, which is further explained in Section III.A. This can include power values or time steps for example. The output layer has the same number of neurons as there are nodes in the FE model (1.5 million). The output of the ANN thus contains the full 3-D temperature distribution on the photonic die. Manual hyperparameter tuning is done for all ANNs in order to find a good trade-off between network size and fitting accuracy. Depending on the training dataset size, the number of hidden layers lies between 2-4 and the number of neurons/layer between 100-1000. For all tested cases, this produced a good fit with the training dataset. The rectified linear activation function (ReLU) is chosen, and the mean squared error (MSE) is the preferred loss function in this case. The network is trained by backpropagation with adaptive learning rate until the loss function converges to a sufficiently small value. A regular ANN is a typical example of black box ML, as the content of the hidden layers is mostly irrelevant and has little physical significance in simulation problems. This can be a disadvantage if the objective is to obtain new insights from the models.

D. POD-RBF-ANN

In this section the proposed hybrid ML method POD-RBF-ANN is explained (Fig.4, right) [21], [24], [28], [29]. Here the input layer takes its original form (as the regular ANN) and the output layer is adapted: the output is no longer a temperature distribution but consists of the POD coefficients. The coefficients are multiplied with the corresponding POD modes and the superposition of all weighted POD modes

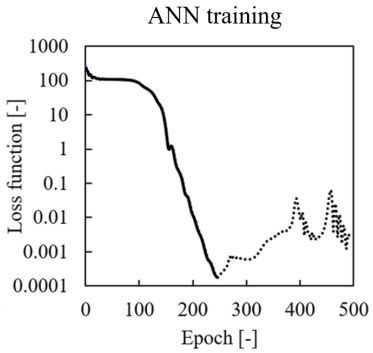

Fig. 5. Network training for ANN

results in a temperature distribution. The advantage here is that the number of neurons in both the input and output layers is greatly reduced compared to the other ANN architectures ($\sim$10 instead of $\sim$millions). Furthermore, this approach uses the best of both methods as it has the capability of non-linear function approximation and regression that is typical for ANNs and the capability of model order reduction through the useage of POD modes. The model training is illustrated in Fig.5: evolution of the loss function (MSE) is shown during the first 500 epochs. The loss function (MSE) for the ANN reaches $<$0.001 after 250 epochs, indicating that a good model fit to the training data is obtained.

III. Design of Experiments

A. Training parameters

The machine learning models require a training dataset which is generated by finite element simulations. The ML models are trained for capturing effects of multiple input parameters:

1) Power dissipation in each channel
2) Die-level boundary conditions
3) Physical channel spacing
4) Time

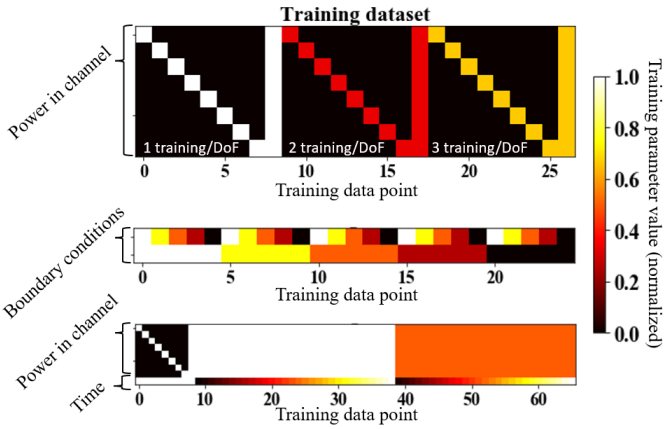

Fig. 6. Matrix heatmap for training input parameters. Each matrix is a visual representation of a training dataset: every column corresponds to a training data point, and every row is a training parameter. color represents the value of the parameter in the specific training data point.

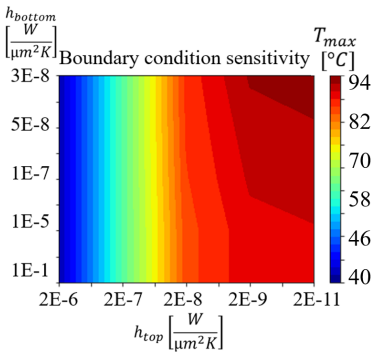

Fig. 7. Temperature contour plot for a wide range of convective boundary conditions for top and bottom side of die (FE result). Results are obtained for a channel spacing of 100 µm, at steady state and nominal power (100%).

These input parameters are selected as they are most influential on the final temperature field. As there are 8 channels in the model, there are 8 degrees of freedom (DoFs) for the power dissipation. The boundary conditions are the thermal cooling conditions on the top and bottom face of the photonics die, so two DoFs. This brings the total DoFs to 12: 11 static input parameters and 1 dynamic (time). The objective is to train the ML models with respect to those input parameters with the smallest amount of training data points as possible. Generating the training dataset takes a significant amount of time (due to the large FE models), which is exactly what has to be avoided.

B. Training dataset design

Now the different input parameters are identified, the training dataset has to be constructed with FE simulations. Careful attention to its design is needed to avoid excessive CPU time in the offline phase. Depending on which input parameter that is being trained, a different amount of training datapoints will be required. Currently, there exist no general rules for training dataset design, as this is very application specific. In this section, it is shown that for multiscale thermal simulation a relatively small training dataset suffices for generating accurate results.

1) Power dissipation: For thermal simulations with constant material properties, different thermal power dissipation levels are considered to follow linear physics. Doubling the thermal power in one channel will results in a double temperature increase. This linear behaviour limits the required amount of data points for power dissipation, as interpolation and extrapolation is straight forward. The design of the training datasets for the power dissipation is shown as a matrix heatmap in Fig.6 (top). The x-axis shows the training data point and the y-axis shows the input parameter number, the color in the matrix represents its value for a specific training point. We start with one training data point per channel and one point where all channels are active simultaneously. This is further expanded to two and three training data points per DoF.

2) Die-level boundary conditions: The maximum temperature in the die is shown in function of the boundary conditions in Fig.7. As the channels are located in the photonic front end of line (FEOL) near to the top side of the die, their temperature is most sensitive to the convective cooling on the (h_{top}). For low cooling configurations (small h_{top}) such as natural convection, the temperature becomes dependent on the boundary condition on the bottom face as well. Both boundary conditions span a large range and are shown on a logarithmic scale. The effect of boundary conditions is non-linear, as shown in Fig.7, and multiple training data points are required to properly capture this effects. For this reason, the full design space is sampled by dividing the whole range of values from Fig.7 in 5 categories for each BC. This results in 25 total unique combinations that are simulated and added to the training dataset (Fig.6, center). The reason why so many training data points are required for only two input parameters is that the effect of one input depends on the value of the other input.

3) Physical channel spacing: The physical channel spacing adds another five training data points. The physical channel spacing will be varied between 60-140 µm in steps of 20 µm.

4) Time: Finally, the time dynamics are also non-linear and one step response is sampled in 29 smaller time steps, as shown in the last dataset in Fig.6. Time series data is typically used to train recurrent neural networks (RNNs), which use the network output of one timestep as input in the next timestep. In this work only standard ANNs are used because they are more generally applicable: not only time as input parameter, bus also all aforementioned input parameters.

C. Test data and accuracy metrics

The prediction or regression accuracy is measured is by comparing the predicted ML result with the ground truth value, which is assumed to be the FE simulation result. With this in mind, extra test data points are generated by FE simulation which are excluded from the training dataset. The accuracy is measured with the mean squared error (MSE) and the relative error (ϵ_{rel}) in each node of the full mesh:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2 \tag{9}$$

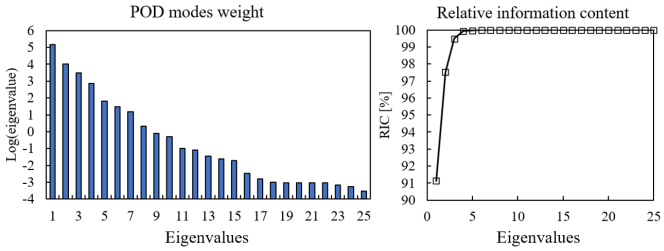

Fig. 8. Eigenvalues for boundary condition training dataset and relative information content (RIC)

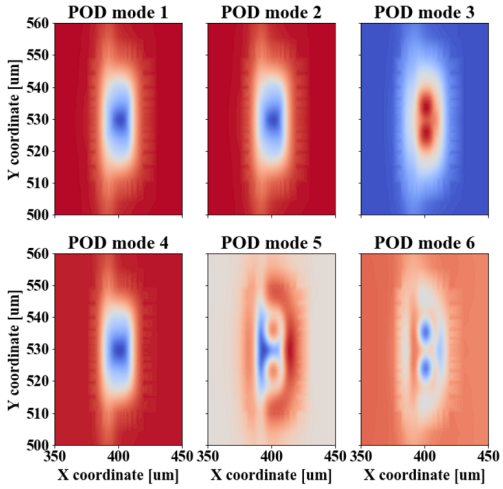

Fig. 9. First 6 POD modes, zoomed in on a single channel

$$\epsilon_{rel} = 100\% \cdot \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \bar{y}_i|}{\bar{y}_i} \tag{10}$$

IV. Results

A. POD modes

After constructing the snapshot matrix (Fig.3) and calculating the SVD, the POD modes are extracted as the columns in the left-singular matrix. From the matrix dimensions it can be derived that the total amount of POD modes is equal to the number of snapshots. However, typically only a few POD modes suffice for capturing 99.9% of the information in the training dataset. The eigenvalues of the different POD modes are shown in Fig.8 on a logarithmic scale. By calculating the relative information content (RIC) according to Eq.5, it becomes clear that the first 4 POD modes already capture approximately all information of the system. Each POD mode is a 3-D map of the full computational mesh. In order to inspect how the POD modes look like, a 2-D top view of the die is made at the location of the photonic devices. Zooming in on one channel, the resulting first 6 POD modes are shown in Fig.9. The first remark is that mode 1,2,4 look very similar. This does not mean that the full POD modes are identical as this is only a zoomed-in image of one specific channel.

B. Validation with test data

The first validation of the ML models is done by comparing the relative error for POD-RBF, ANN and POD-RBF-ANN

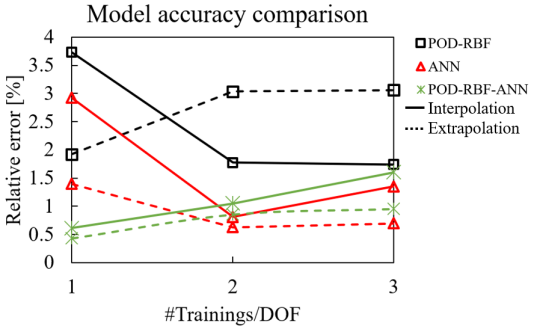

Fig. 10. ML prediction accuracy vs. training dataset size, for interpolation and extrapolation

for the linear training data. The models are trained for 8 different input parameters: the power dissipation in the eight different channels (Fig.6). First one training data point per DoF is used, then two and finally three. The results are shown in Fig.10 for both interpolation and extrapolation predictions. The input power in the training dataset is normalized between 0-1, so interpolation is by definition inside this range and extrapolation is >1. The first conclusion is that for none of the cases there is an accuracy increase by extending the training dataset from two trainings/DoF to three trainings/DoF. Going for two trainings/DoF instead of one has mixed results. In some cases this also decreases the accuracy due to overfitting. In general, for linear dynamics, very few training points/DoF are sufficient for accurate results. POD-RBF-ANN performed best, followed by ANN and finally POD-RBF.

The next input parameter which is investigated is the time. Even though this is only one degree of freedom, many training data points are required for capturing the time dynamics of the thermal step response as this is typically an exponential function. The ML prediction vs. the ground truth (FE) is shown in Fig.11 for ANN and POD-RBF-ANN. POD-RBF failed to properly simulate the time dynamics with the given training dataset. The different colors represent the relative power per channel: 0-50-100% are in the training dataset and 25-75-125-200% are not. In general, a good match between ML and FE is obtained for all cases (training and tests), except for the simulation at 200% there is large error. The simulation of different channel spacings is equally accurate for all methods. The 1-D temperature profile for five different spacings simulated by ANN is shown in Fig.12.

As a next step the ML models are trained with respect to the boundary conditions. A test dataset consisting of five random combinations of boundary conditions is generated and the accuracy of the ML models is benchmarked in Fig.13. Both the MSE and relative error is shown. On average, for all models the relative error lies between 1-2%. POD-RBF-ANN performed best, followed by ANN and finally POD-RBF which has one outlier with large error for data point 2.

The logical final step is to combine all training datasets into one big dataset that is used to train all ML models. The

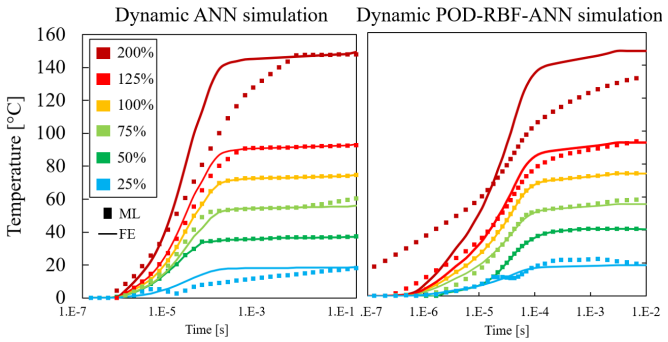

Fig. 11. Transient simulation comparison between ANN and FE, 100% and 50% power are part of training dataset

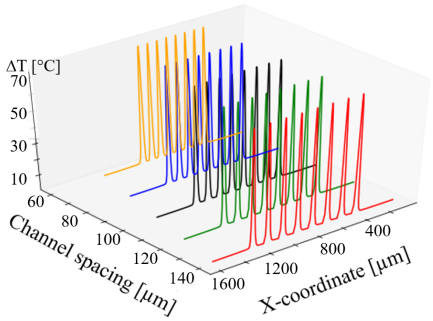

Fig. 12. Temperature profile prediction (ANN) for different channel spacings

heatmap of the training parameters is displayed in Fig.14. All static inputs are combined, so the time is left out. In total there are 39 training data points for 11 different input parameters. Furthermore, three test data points are generated with randomized values for all input parameters. The prediction accuracy of ANN and POD-RBF-ANN is shown in Fig.15. POD-RBF is not included as the RBF method was unable to produce useful results for the combination of multiple non-linearities. Both ANN and POD-RBF-ANN prediction accuracy is comparable, with relative error ∼2%. The temperature profile across all eight channels for one of the test data predictions with ANN is shown in Fig.16, showing a good match with the FE reference data.

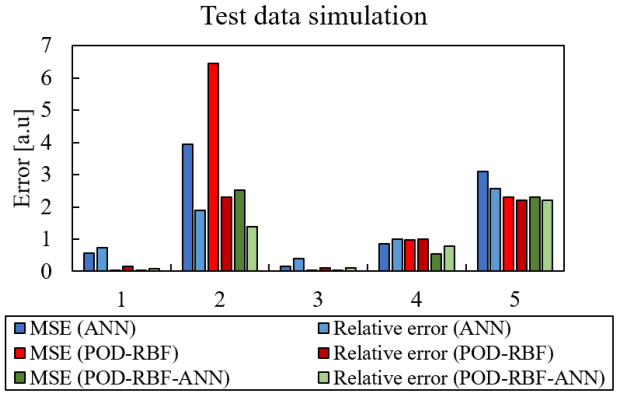

Fig. 13. ML accuracy verification on randomized test dataset for different boundary conditions

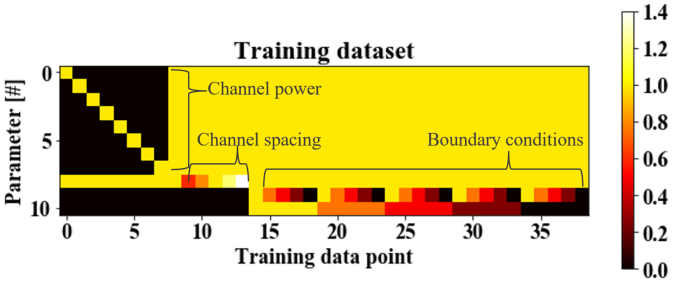

Fig. 14. Combined training dataset for all static parameters

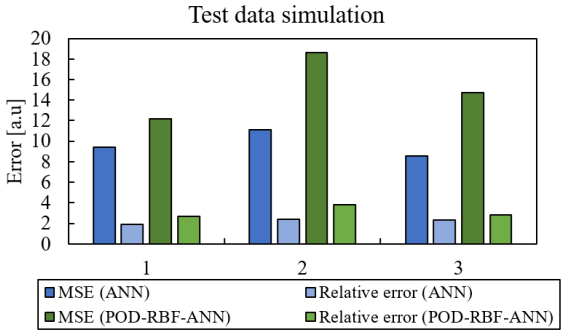

Fig. 15. ML accuracy verification on randomized test dataset for all (static) input parameters

C. Simulation speed increase

The simulation speed increase compared to the reference FE simulations is now discussed. More specifically, the phases in the simulation process are compared between the different methods (FE and ML): the online and offline phase. The online phase is defined as the phase during which a new simulation result is generated, given a set of input parameters. For FE simulation, the online phase is all CPU time required for solving the system. For the ML methods, there is a great time requirement for training: first there is training data generation, then processing of the dataset (e.g. SVD) and finally model fitting. All time required for the aforementioned training is categorized under the offline phase of the simulation process. All calculations are performed on a Intel(R) Core(TM) i5-10310U CPU @ 1.70GHz processor with 16 GB RAM. In Fig.17 a cross section of the die is shown, close to the top side. The solution is zoomed in

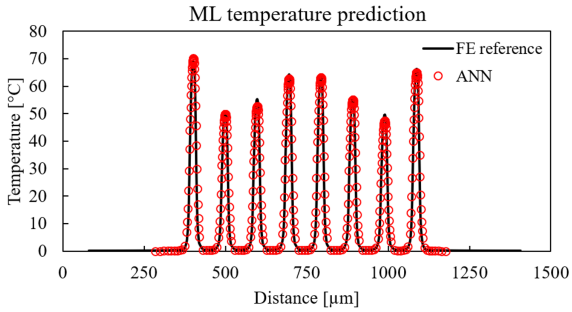

Fig. 16. ANN temperature profile prediction for test data point (3) vs. ground truth (FE reference)

TABLE II
CPU TIME BENCHMARKING OF DIFFERENT ML METHODS

Phase		ML methods for multiscale simulation			
		FE	**ANN**	**POD-RBF**	**POD-RBF-ANN**
Offline phase	Training data generation by FE [min]	n.a.	472	472	472
	SVD [s]	n.a.	n.a.	71	71
	Model fitting [s]	n.a.	513	n.a.	8
	Total (with training) [min]	0	8.55 (480.6)	1.18 (473.2)	1.32 (473.3)
Online phase	Simulation [s]	726	0.21	10.83	0.096

TABLE III
TIME COMPLEXITY AND CPU TIME SCALING (ONLINE PHASE) WITH THE NUMBER OF NODES

	Number of nodes		
CPU time [s]	$2.5 \cdot 10^5$	$1.5 \cdot 10^6$	Increase
FE	24	726	X30.3
ANN	0.057	0.21	X3.7
POD-RBF	1.652	10.83	X6.6
POD-RBF-ANN	0.029	0.096	X3.3

on two active channels. The reference CPU time of the online phase is 726 seconds for the FE simulation. Going to POD-RBF this is reduced to 10.83 seconds, for ANN this is 0.21 seconds and for POD-RBF-ANN this is 0.096 seconds. This translates into a computational speed increase between X67-X7500. The reason for the longer CPU time of POD-RBF is that the RBF interpolation requires solving a system of equations by matrix inversion (see Eq.8). Here the potential performance gain of using ML in combination with FE becomes clear: significantly faster simulations that allow for quicker analysis and insight into the models. In order to obtain the very fast ML models, an extended offline phase is required. This additional time and effort for generating the training dataset must be considered when deciding on whether or not the ML models are useful.

The overview of the different offline phase times is shown in Table II. The training data generation is for taken for the case with 39 data points. It becomes clear that for all ML methods the offline phase is dominated by the training data generation, highlighting the importance of minimizing the number of data points in the training dataset. Comparing the offline phase between the different ML methods (without training data generation), it is concluded that POD-RBF is the fastest as no model fitting is required for this method. This is followed by POD-RBF-ANN where model fitting is necessary, but extremely fast as the input and output layer dimensions of the ANN are limited. Finally, the regular ANN requires the longest offline phase because of the extended model fitting time, caused by the large number of tuneable parameters.

The time CPU time scaling with problem size is now further investigated. In order to do this, a FE model with a smaller number of nodes is created (see Section V.A). The finite element solver employs a sparse, multi-frontal direct solver which has a typical time complexity of $\mathcal{O}(n^2)$. This scaling is confirmed by numerical experimentation: for a mesh with $2.5{\cdot}10^5$ nodes the CPU time is 24 seconds, and for a mesh with $1.5{\cdot}10^6$ the CPU time is 726 seconds, an increase with factor X30.25 (see Table III). This value is close to the theoretical scaling X36. The time complexity of ANNs depends on the network architecture: the number of layers and nodes per layer. For a fair comparison, the network architecture is kept the same, except the size of the output layer, which scales with the number of nodes in the FE model. This implies a time complexity of $\mathcal{O}(n)$ as the number of required operation scales linearly with the output layer size. This is confirmed through numerical experimentation: going from small to large mesh, the CPU time increase from 0.057 seconds to 0.21 seconds, an increase with factor $\sim$4. One side note here, is that because of the extremely small CPU times for the ANN online phase, it is difficult to determine the exact required time, because there is variability in the recorded CPU time that depends on background processes. The time complexity of POD-RBF-ANN shows the same trends as ANN. For the POD-RBF algorithm, the online phase consists of matrix inversion in order to determine the POD coefficients. Matrix inversion itself has a time complexity of $\mathcal{O}(n^3)$, however in this case n is not the number of nodes, but the number of POD modes. Because this number is very small in the tested cases ($n = 4$), the CPU time of the online phase for POD-RBF is mainly memory-limited because the size of each POD modes is equal to the number of nodes. This implies that the time complexity of POD-RBF with respect to n the number of nodes is $\mathcal{O}(n)$, and this is confirmed experimentally: the CPU time increased from 1.652 seconds to 10.83 seconds. In conclusion, direct solvers for finite element problems have a time complexity of $\mathcal{O}(n^2)$ where n is the number of nodes, while the tested ML methods have a time complexity of $\mathcal{O}(n)$. This shows that the larger the computational mesh (and hence number of nodes), the more time saving potential there is for the ML models in the online phase. Generating the training dataset through FE simulation will take more time for models with a large number of nodes, which needs to be taken into account.

V. CASE STUDIES

A. Compact model for photonic circuit design

The next aspect that is discussed is a possible application for a fast multiscale thermal model for photonics. It has been proven that ML models trained with FE data show great potential for computational time reduction and are sufficiently accurate. The integration of a compact ML model in a photonic integrated circuit (PIC) design tool opens up the opportunity to assess the thermal coupling or crosstalk between temperature sensitive devices. This is a notorious problem for PICs and

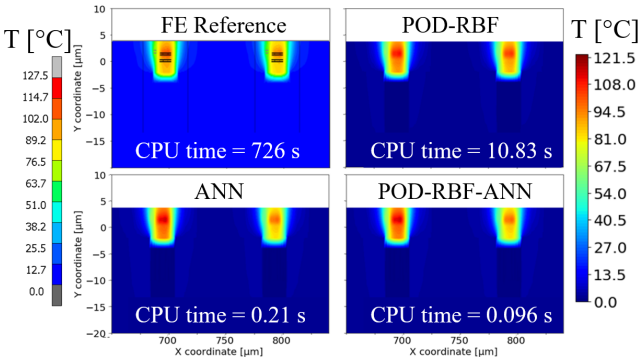

Fig. 17. Temperature simulation results, vertical cross section of die top-side is shown. Left colorbar is for FE reference, right colorbar is for ML results.

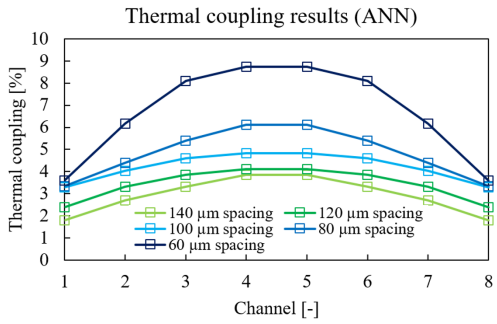

Fig. 18. Thermal coupling in function of channel number, with channel spacing as parameter

due to the high thermo-optic coefficient of the Si waveguide the devices are temperature sensitive [12]. Thermal coupling in each channel is calculated for a series of 8 channels by activating each channel separately, followed up by activating all channels simultaneously. The coupling is very sensitive to boundary conditions, the spacing between the channels and the location of the channel within the filter array. In Fig.18 the relative thermal coupling is expressed in function of the channel number, with spacing as parameter. All data points are calculated with ANN, using the training dataset design rules derived in Section III.D. Because the thermal response to changing the physical channel spacing is non-linear, 6 training datapoints are used for training with respect to this single input parameter. The choice for ANN is arbitrary, POD-RBF and POD-RBF-ANN both produce similar results in this case. Firstly, channels at the center (4-5) become hotter due to their close proximity to all other channels. Secondly, increasing spacing from 60 µm to 140 µm greatly reduces the overall thermal coupling across the array. In total 45 ML simulations were required for obtaining this result (9 simulations/spacing and 5 spacings). By using the compact ML model, these data are generated in a couple of seconds, compared to 544 minutes (∼10 hours) with FE simulation.

B. Compact model for package thermal simulation

In order to compare the performance of the proposed machine learning tools with the HotSpot and Power Blurring method, the Bobcat mobile processor test case introduced in [7], [17] has been used. A thermal finite element model has been generated for this geometry, which is used as a reference solution, and from which the training data for the ML models has been derived. For the comparison, the results on the HotSpot and Power Blurring simulations have been used from [7]. The processor is modelled as a solid Si block, flip-chipped on a heat spreader substrate. The heat spreader is connected to a heat sink with low thermal resistance ($R = 1\ K/W$). On the interfaces between the different materials there is a thermal interface material (TIM) present. The finite element model and schematic cross-section are shown in Fig.19 (a,b). All details about material properties and geometry can be found in [7]. The power map of the processor is shown in Fig.19 (c). With the purpose of training the ML models, the power in four active processor tiles is used as input parameter. A fifth parameter is the background power. In total there are five input parameters. A small training data set is generated by activating each parameter separately, and once all together: in total only six training simulation are done. A temperature contour plot for the simulation with reference power map (Fig.19 (c)) is shown in Fig.20. A horizontal cross-section through the Si die is taken at the location of the active processor front side. Two test cases are defined which are not included in the training data set. The values of the input parameters for each case are shown in Fig.21. Test case (a) is conservative in the sense that the values do not deviate far from the training data. Test case (b) has a much bigger difference between training and test. Test case (b) is mainly added with the purpose of highlighting the limitations of the ML models, which typically lose their accuracy outside the bounds of the training data set.

The temperature profile (A-A' in Fig.20) for case (a) and case (b) is shown in Fig.22. The (assumed) ground-truth, the finite element simulation, is the reference that is used for error calculation. In Fig.22 it can be seen that for case (a) all ML methods perform very well. In case (b), there is a clear discrepancy visible between the ML methods and the FE result. POD-RBF-ANN performs best out of the three ML methods. In Fig.22 (c) the ML results are compared with the results of HotSpot and Power Blurring [7]. The total performance comparison between all methods is shown in Table IV. Here, the performance of HotSpot and Power Blurring is added for the same test case. The data for both these methods is obtained from [7]. The CPU time for all methods is $<$1 second for the online phase. The accuracy of the ML methods is competitive with the HotSpot and Power Blurring result. However, a very important conclusion here is that for the second simulated case the ML methods perform significantly worse compared to HotSpot and Power Blurring. Average error is ∼10% and max error is ∼50%. The reason for this is that a very small training data set (6 data points) is used for training the ML models and any significant extrapolation outside of the training bounds of the input parameters will result in poor model performance. To summarize the comparison between ML, HotSpot and Power Blurring, the (dis)advantages of each method are highlighted below:

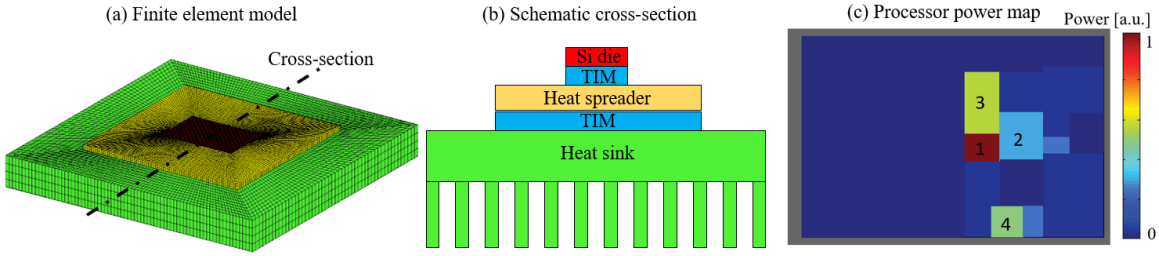

Fig. 19. Bobcat mobile processor thermal model: (a) finite element model, (b) schematic cross-section and (c) normalized processor power map

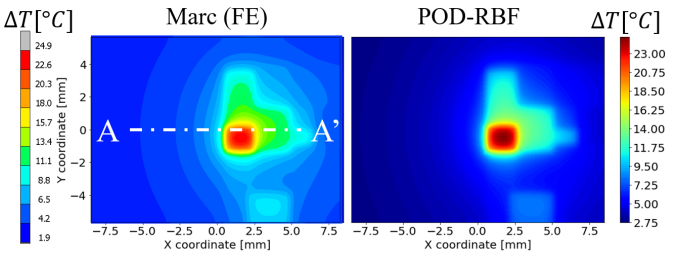

Fig. 20. Temperature contour plots for FE and POD-RBF simulation results of the reference power map. A-A' shows the line for temperature profile extraction.

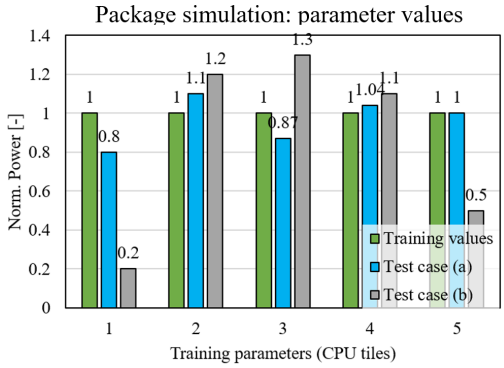

Fig. 21. Normalized power in five active processor tiles: training data is obtained with the reference power values (Fig.19 (c)). Case (a) is a conservative test data point, only a small deviaition bewtween test and training is used. Case (b) is more aggressive.

- *Machine Learning:*
  1) Time complexity: $\mathcal{O}(n)$
  2) Advantages: can be used for different boundary conditions (if included in the training data). Based on FE: you can include very complex 3D geometries for the heat sources.
  3) Disadvantages: the accuracy of the method depends on the proximity between the evaluated parameter set and the training data. Generation of training dataset can take considerable amount of time.
- *Power Blurring:*
  1) Time complexity: $\mathcal{O}(n \cdot log(n))$
  2) Advantages: fast and efficient calculation due to convolution in space and time. Exact solution of heat equation in the Si die itself. Very good accuracy for heat sources in the central area of the chip. Method usable for variable chip sizes. Very suitable to explore different power dissipation patterns.
  3) Disadvantages: Unit cell thermal response ('mask') needs to be recalculated in FE for each boundary condition and package value. Thermal spreading in chip package and heat spreader not included in the model. This leads to an overestimation of the chip temperature at the edges. Limited to a single material property per horizontal layer in the considered geometry (Green's function limitation)
- *HotSpot*
  1) Time complexity: $\mathcal{O}(n^2)$
  2) Advantages: actual calculation of the temperature distribution for a discretized RC network: allows variations of power distribution as well as all material properties and dimensions.
  3) Disadvantages: Not as fast as other methods. Furthermore, the accuracy depends on the resolution of the discretization. Current version does not yet include option to allow different materials per layer: this is acceptable for planar heat source in Si.

VI. Conclusion

Multiscale thermal simulation of integrated (photonic) circuits is computationally expensive with traditional finite element models. This is the motivation for benchmarking the performance of multiple machine learning algorithms using the same training dataset for speeding up the simulation process. Artificial neural networks (ANN) are useful for simulating non-linear dynamics but are black box and their interpretability is low. Proper orthogonal decomposition combined with radial basis functions (POD-RBF) is a method where fundamental behaviour and patterns are extracted from the training data. The strength of this approach are the highly interpretable POD modes, but the drawback is that non-linear dynamics are poorly captured when using a minimal training dataset design. Furthermore, a hybrid ML method is tested: POD-RBF-ANN, where the output layer of the ANN is used for predicting the POD mode coefficients. This greatly reduces the ANN network size and number of tuneable parameters. Prediction accuracy with a relative error of $\sim$3% is obtained for the test data and computational speed increase between X67-X7500 is shown. To conclude, FE-trained ML models can potentially be integrated in circuit design tools in order to efficiently assess the thermal performance of a layout.

TABLE IV
BENCHMARKING OF DIFFERENT ML METHODS AGAINST STATE-OF-THE-ART METHODS FOR PACKAGE THERMAL SIMULATION. THE RESULTS MARKED WITH * ARE OBTAINED FROM [7]. ML RESULTS ARE SHOWN FOR CASE (A) AND CASE (B) BETWEEN BRACKETS.

	FE	POD-RBF	ANN	POD-RBF-ANN	Hotspot*	Power Blurring*
Computation time [s]	24	1.652	0.057	0.029	0.11	0.041
Error in hot-spot [%]	-	5.92 (55.8)	5 (52.9)	1.8 (19.1)	12.9	0.14
Max error [%]	-	5.92 (55.8)	5 (52.9)	1.8 (19.1)	25.7	13.7
Avg. Error [%]	-	1.98 (11.8)	1.6 (13.1)	0.7 (4.1)	6.5	2.5
Abs. Error range [°C]	-	1.13 (5.67)	0.95 (5.38)	0.3 (1.93)	4.2	0.56

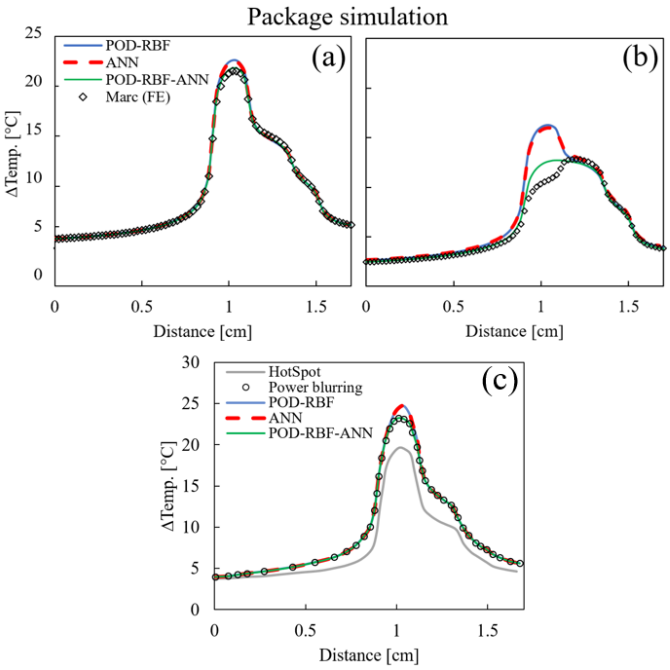

Fig. 22. Simulation result: temperature profile (A-A') through Si die. Case (a) is similar to training data, while case (b) has a much different power map. In (c) the ML results are compared with HotSpot and Power Blurring, reproduced from [7]

	Finite element	ANN	POD-RBF	POD-RBF-ANN
Accuracy	~100%	~97%	~97%	~97%
Simulation time	726 s	<1 s	<10 s	<0.1 s
Training time		513 s	0 s	8 s
Linear dynamics		✓	✓	✓
Non-linear dynamics		✓	<94%	✓
Combination of non-linearities		✓	<50%	✓

Fig. 23. Performance scorecard for ML methods for multiscale thermal simulation tested in this work. Dark green = best, red = worst. Numbers shown are relative accuracy (%) and simulation, training time (s)

ACKNOWLEDGMENT

This work was carried out as part of imec's industry affiliation R&D program 'Optical I/O' and supported by imec's expertise center AR2T (Advanced Reliabilty, Robustness and Test). The implementation of POD-RBF in Python is made possible by the open-source code 'EZyRB' developed by Marco Tezzele and Nicola Demo from Mathlab. The implementation of the artificial neural networks is done with the open-source Python libraries TensorFlow and Keras.

REFERENCES

[1] L. Pavesi, D.J. Lockwood, "Silicon Photonics III: Systems and Applications," *Springer*, 2016.
[2] H. Oprins, Y. Ban, V. Cherman and J. Van Campenhout, "Thermal Aspects of Silicon Photonic Interposer Packages," *EuroSim Conference*, 2020.
[3] B. Barabadi et al., "Multiscale Transient Thermal Analysis of Microelectronics," *Journal of Electronic Packaging (ASME)*, Vol.137, 2015.
[4] Z. Yuan et al., "PACT: An Extensible Parallel Thermal Simulator for Emerging Integration and Cooling Technologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol.41, No.4, pp.1048-1061, April 2022.
[5] *HotSpot Thermal Simulator*, [online], Available: http://lava.cs.virginia.edu/HotSpot/index.htm.
[6] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron and M. R. Stan, "HotSpot: a compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.14, No.5, pp.501-513, May 2006.
[7] A. Ziabari et al., "Power Blurring: Fast Static and Transient Thermal Analysis Method for Packaged Integrated Circuits and Power Devices," *IEEE Transactions on very Large Scale Integration (VLSI) Systems*, Vol.22, No.11, pp.2366-2379, Nov. 2014.
[8] F.L.T. Maggioni, H. Oprins, E. Beyne, I. De Wolf, M. Baelmans, "Fast convolution based thermal model for 3D-ICs: Methodology, accuracy analysis and package impact," *Microelectronics Journal*, Vol.45, No.12, pp.1746-1752, 2014.
[9] X. Wang and S. Mookherjea, "Fast circuit modeling of heat transfer in photonic integrated circuits," *CLEO Conference*, 2017.
[10] T.Y. Wang and C.C.P. Chen, "SPICE-Compatible Thermal Simulation with Lumped Circuit Modeling for Thermal Reliability Analysis based on Modeling Order Reduction," *IEEE International Symposium on Quality Electronic Design*, March 2004.
[11] D. J. Walkey et al., "Compact, Netlist-Based Representation of Thermal Transient Coupling Using Controlled Sources, "*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol.23, No.11, pp.1593-1596, Nov. 2004.
[12] D. Coenen et al., "Analysis of Thermal Crosstalk in Photonic Integrated Circuit Using Dynamic Compact Models," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, Vol.12, No.8, pp.1350-1357, Aug. 2022.
[13] Jedec Solid State Technology Association, "Transient Dual Interface Test Method for the Measurement of the Thermal Resistance Junction to Case of Semiconductor devices with Heat," 2010.
[14] F. L. T. Maggioni, H. Oprins, E. Beyne, I. De Wolf and M. Baelmans, "Fast Transient Convolution-Based Thermal Modeling Methodology for Including the Package Thermal Impact in 3D ICs," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, Vol.6, No.3, pp.424-431, March 2016.

[15] V. Sodan, F. L. T. Maggioni, H. Oprins, S. Stoffels, M. Baelmans and I. De Wolf, "New fast distributed thermal model for analysis of GaN based power devices," *2016 46th European Solid-State Device Research Conference (ESSDERC)*, 2016, pp. 172-175.

[16] F. Maggioni, H. Oprins, E. Beyne, I. De Wolf and M. Baelmans, "Convolution based compact thermal model application to the evaluation of the thermal impact of die-to-die interface including interconnections," *Fourteenth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, 2014, pp. 98-106.

[17] B. Burgess at al., "Bobcat: AMD'S Low-Power X86 Processor," *IEEE Micro*, Vol.31, No.2, pp.16-25 Jan. 2011.

[18] H. Oprins et al., "Thermal analysis of 3D functional partitioning for high-performance systems," *ITherm Conference*, pp.145-153, 2020.

[19] D. Coenen, H. Oprins, I. De Wolf, "Circuit-level Thermal Modelling of Silicon Photonic Transceiver Array using Machine Learning," *ITherm Conference*, 2022 (accepted).

[20] K. Sikka et al., "Artificial Neural Networks for Package Thermal Analysis," *ITherm Conference*, 2021.

[21] O.M. Agudelo, J.J. Espinosa and B. De Moor, "Acceleration of nonlinear POD models: A neural network approach," *Proceedings of the European Control Conference*, 2009.

[22] A. Sridhar et al., "Neural Network-Based Thermal Simulation of Integrated Circuits on GPUs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 31, No.1, pp.23-36, Jan. 2012.

[23] L. Jiang et al., "An Effective and Accurate Data-Driven Approach for Thermal Simulation of CPUs," *ITherm Conference*, 2021.

[24] O. San et al., "An artificial neural network framework for reduced order modeling of transient flows," *Communications in Nonlinear Science and Numerical Simulation*, Vol.77, pp.271-287, 2019.

[25] M.C. Cheng et al., "POD-based Thermal Model for FinFET IC Structure," *ICST Conference*, 2014.

[26] Y. Ye, W. Zhang, W. Liu, "Thermal-Aware Design and Simulation Approach for Optical NoCs,"*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 39, No.10, pp.2384-2395, Oct. 2020.

[27] Y. Ye et al., "System-Level Modeling and Analysis of Thermal Effects in WDM-Based Optical Networks-on-Chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 33, No.11, pp.1718-1731, Nov. 2014.

[28] H.F. Lui, W.R. Wolf, "Construction of Reduced Order Models for Fluid Flows Using Deep Feedforward Neural Networks," *Journal of Fluid Mechanics*, Vol.872, pp. 963-994, Aug. 2019.

[29] J.S. Hesthaven, S. Ubbiali, "Non-intrusive reduced order modeling of nonlinear problems using neural networks," *Journal of Computational Physics*, Vol. 363, pp.55-79, Feb. 2018

[30] N. J. Molinaro, "The Two Point Correlation Structure of a Cylinder Wake," *M.Sc. thesis Virginia Polytechnic Institute and State University*, 2017.

[31] W. Jia et al., "Fast Thermal Simulation of FinFET Circuits Based on a Multiblock Reduced-Order Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol.35, No. 7, pp.1114-1124, July 2016.

[32] M. Rakowski et al., "Hybrid 14nm FinFET - Silicon Photonics Technology for Low-Power Tb/s/mm2 Optical I/O," *Symposium on VLSI Technology Digest of Technical Papers*, 2018.

[33] MSC Software Corporation, *Marc: Advanced Nonlinear Simulation Solution*, accessed on 5/04/2022, [Online] Available: https://www.mscsoftware.com/product/marc.

[34] D. Coenen et al., "Thermal modelling of Silicon Photonic Ring Modulator with Substrate Undercut," *IEEE Journal of Lightwave Technology* , 2022 (early access).

[35] Demo et al., (2018). "EZyRB: Easy Reduced Basis method." *Journal of Open Source Software*, 3(24), 661, 2018.

[36] B. Barabadi et al., "Transient Heat Conduction in On-Chip Interconnects Using Proper Orthogonal Decomposition Method," *Journal of Heat Transfer (ASME)*, Vol.139, 2017.

[37] V. Buljak, G. Maier, "Proper Orthogonal Decomposition and Radial Basis Functions in material characterization based on instrumented indentation," *Engineering Structures*, Vol.33, pp.492-501, 2011.

[38] W. du Toit, "Radial Basis Function Interpolation," *M.Sc. thesis University of Stellenbosch*, 2008.

[39] F. Chollet et al., *Keras,* accessed on 2/05/2022, [online] Available: https://github.com/keras-team/keras

[40] M. Abadi et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,* accessed on 2/05/2022, [online] Available: https://github.com/tensorflow