FEW-GROUP CROSS SECTIONS MODELING BY ARTIFICIAL NEURAL NETWORKS

Szames E., Ammar K., Tomatis D. and Martinez J.M.
CEA, DEN, DM2S, Service d'études des réacteurs et de mathématiques appliquées,
Université Paris-Saclay F-91191 Gif-sur-Yvette, France

esteban.szames@cea.fr, karim.ammar@cea.fr,
daniele.tomatis@cea.fr, jean-marc.martinez@cea.fr

ABSTRACT

This work deals with the modeling of homogenized few-group cross sections by Artificial Neural Networks (ANN). A comprehensive sensitivity study on data normalization, network architectures and training hyper-parameters specifically for Deep and Shallow Feed Forward ANN is presented. The optimal models in terms of reduction in the library size and training time are compared to multi-linear interpolation on a Cartesian grid. The use case is provided by the OECD-NEA Burn-up Credit Criticality Benchmark [1]. The Pytorch [2] machine learning framework is used.

KEYWORDS: Homogenized Cross Sections, Machine Learning, Artificial Neural Networks, Supervised Learning

1 INTRODUCTION

Few-group cross sections come from transport calculations where the neutronic flux is computed using a detailed discretization in energy and space. These are real valued scalar functions obtained through an homogenization process over a d-dimensional rectangular domain, which can be mapped to the unit hypercube $\mathcal{X} = [0,1]^d$. Lattice calculation points deliver the few-group cross section set $\mathcal{Y} = \{\sigma_{irg}(x) \rightarrow \mathbb{R}, x = (x_1, \ldots, x_d) \in \mathcal{X}\}$ for each specialized isotope i, of the decay chain used by the core solver, reaction type r and group g. Cross sections usually present smooth profiles with possibly strong variations in localized regions and low order dependence among the variables. The modeling goal consists in finding the approximations $\hat{\sigma}_{irg} \simeq \sigma, \forall \sigma \in \mathcal{Y}$.

The transport code APOLLO2.8 [3] was used to generate cross section data for the OECD-NEA Burn-up Credit Criticality Benchmark, whose material and geometrical specifications are fully available in [1]. It consists of a typical PWR fuel assembly composed of 17×17 UO_2 fuel rods with 4% w/o ^{235}U enrichment and with 25 guide tubes. Spatial homogenization is performed over the whole assembly with energy condensation to the thermal and to the fast groups (cutoff energy at 0.625 eV). We analyze the cross sections that capture the majority of the macroscopic cross section behavior (high ratio $\sigma_{irg} C_i / \sum_i C_i \sigma_{irg}$ being C_i the concentration of the i-th isotope).

2 MODELING CHALLENGES AND STATE OF THE ART

Ordinary reactor studies for fuel cycle optimization, transient simulations and control analysis need to resolve the multi-physics coupling with other codes where cross sections models are dealing with ever growing volumes of data. Many industry applications approximate homogenized cross sections by a first order piece-wise polynomial interpolation, here called Multi-Linear (ML), usually

adopting a Cartesian sampling rule for $\mathcal{X}$ [4]. In this schema data are simply stored to be fastly interpolated on demand. However, the number of data points may grow exponentially with increasing dimensions issuing the "Curse of Dimensionality" [5] being the library size $|\mathcal{Y}| \times \prod_{i=1}^{d} \mathcal{X}_i$ with $\mathcal{X}_i$ the coordinates in the i axis of the samples. Many points may be necessary to achieve accurate reconstruction with target cross section relative errors laying between $1\mathrm{E}-1$ and $1\mathrm{E}-2$ [6]. Classical techniques such as higher order approximation spaces, projection into sub-libraries or regression with global polynomials [5] have been used to address this problem, though incurring sometimes in high localized errors or polynomial degrees [7]. Other works have focused instead on a smarter sampling of $\mathcal{X}$ by Active Learning procedures that select the most informative points [8], or by Sparse Grid rules that may impose nonetheless additional constrains on the functional space used [6]. Effort has been put forward to face these interrelated challenges through methodologies that try to avoid a priori assumptions on the approximator's structure, while still finding a tailored $\mathcal{X}$ for the emerging model [9].

ANN constitute an alternative to these methods, allowing to approximate functions without formulating explicit relations among the variables. In regression problems they are sometimes referred to as black-box universal approximators [10]. ANN have been applied in a broad range of nuclear engineering problems such as core parameter prediction and control, fuel management optimization, transient diagnosis and inventory estimation [11]. Yet, the design of an optimal ANN remains a difficult iterative process due to the high amount of free parameters, the wide range in which they can vary and the strong inter-dependence among them. Few works can be found in litterature on cross section modeling by ANN. Good performances were obtained in [12] but using simple architectures in 1-dimensional domains. We will focus on a *sensitivity analysis* for the main aspects of ANN modeling applied to cross sections, demonstrating its applicability to an industry case and discussing its performance. Each cross section is approximated by an independent ANN, postponing more complex network designs with several outputs for instance, for future studies.

3 ARTIFICIAL NEURAL NETWORKS

ANN are computing systems composed of processing elements called neurons that, connected to each other, emulate their biological counterparts. In a Feed Forward ANN the information travels forward through the layers, thus the name. They are called *Shallow* ANN with only one layer and *Deep* ANN with more. The strength of the connections between neurons are called weights and we consider only neurons that are *fully connected* between layers. Non-linear *activation functions* f complete the output of the neurons from a given layer enabling the network to learn complex patterns. A layer considers a linear combination in the form $f(W_i z + b_i)$ with the biases $b_i \in \mathbb{R}$ and the vector of weights $W_i \in \mathbb{R}^{N_i}$) for N_i neurons of the layer i. For L layers the output of an ANN is $\hat{\sigma} = f^L \circ f^{L-1} \circ \cdots \circ f^1(x)$, $x \in \mathcal{X}$. The ANN parameters are determined by a training process which is a supervised learning task, based on the minimization of a *loss function* $\mathcal{L}$. A training set provides the samples used in a forward pass to calculate $\hat{\sigma}$, thus allowing to estimate the gradients in order to minimize $\mathcal{L}(\sigma, \hat{\sigma})$. The gradients modulated by the *learning rate* α are used to update the weights and the biases in a computationally convenient back-propagation pass. The approximation power of an ANN lays in the network capability to adapt without imposing prior task-specific rules by a training process that may become computationally expensive. It is customary to divide the training data into batches and an iteration on the whole training set completes an *epoch*.

Even for a classical Feed-Forward ANN the amount of free parameters to consider is rather large. The design of the ANN requires to define the amount of hidden layers, neurons per layer, activation

function, parameter's initialization and possible regularization. Preprocessing of data is mandatory. Training hyper-parameters must be chosen as well: learning rate, number of batches, loss function with possible regularization, etc. This inter-dependent corpus of choices defines the sensitivity studies to perform.

4 RESULTS

The Pytorch package was used [2], and our script is available online under MIT license [13]. The model's accuracy is set forth in terms of the mean average error ME $= \sum_{i=1}^{M} abs(\hat{\sigma}_i - \sigma_i)/M$, and the mean average relative error MRE $= \sum_{i=1}^{M} 100abs(\hat{\sigma}_i/\sigma_i - 1)/M$ being M the size of the set. The model's relative error over all cross sections is TE $= \sum_{i=1}^{|\mathcal{Y}|} \text{MRE}_i/|\mathcal{Y}|$.

We consider the linearly independent variables: the Burnup (0 - 45 GWd/t), the Fuel Temperature (500 - 1000 °C) and the Boron Concentration (300 - 800 ppm) sampled in a $170 \times 16 \times 16$ grid totalizing 43520 data points. The number of dimensions is thus $d = 3$ and the data set is divided in a randomly sampled *Train*/*Test* split of 80/20 %. Train data are further divided in 5 randomly sampled batches aiding the optimizer (Adam [2]) to exit local minima, and so 5 network adaptations occur per epoch. Test data is reserved to check the generalization capability of the model.

NORMALIZATION AND TRAINING HYPER-PARAMETERS

Sensitivity analysis were performed by varying relevant model parameters independently. Following literature recommendations we considered a Shallow ANN with number of neurons $N = 20$ and activation function $f(x) = tanh(x)$ due to its good mapping capabilities, zero mean and smooth output [11]. Test MRE at 1E5 epochs is presented in Tables 1, 3, 4 for the cross section $\text{U}_{f,2}^{235}$ and $\Sigma_{a,1}$ where the optimal parameters are highlighted in green. In each study the parameters which are not varied remain in green. *Data normalization* has a strong effect in the gradient's absolutes values and thus in the overall training process. Different combinations for $\mathcal{X}$ and $\mathcal{Y}$ are presented in Table 1. Without cross section normalization the optimizer is unable to converge with errors of 100%. Normalizing by $max(\sigma(x))$ bounds cross section values to 1. $\mathcal{X}$ is already normalized to the unit hypercube and errors are not decreased further when considering only for the burnup $\sqrt{\text{Bu}}$. Using $log(\sigma)$ and zero mean did not improve the results in a significant way. Normalizing by the variance on the other hand, exhibits the best results. Different initializations of the weights using Random, Normal, Uniform or Xavier-Uniform distributions (see [2]) were analyzed with negligible impact in the MRE and $\mathcal{L}$ profiles. The loss functions considered are presented in Table 4, and SmoothL1Loss is detailed in [2]. With respect to the MRE, the L_1 loss function showed the best results, as expected with respect to the MRE error. The activation function affects the convergence during training and the evaluation cost of the ANN. Results with different activation functions are reported in Table 3.

The learning rate α modulates the actualization of the weights influencing the convergence rate and the optimizer's possibility of escaping local minima. If it is too high the algorithm may not converge and if it is too low it may do so too slowly. This behavior is confirmed: for $\alpha = 0.1$ errors fluctuated up to 2 orders of magnitude around a MRE of $5\text{E} - 1$ while for $\alpha = 0.00001$ it converged smoothly but very slow. A value of $\alpha = 0.001$ is chosen as a good compromise. Different numbers of batches of 1, 5, 100 and 1000 were tested. A single batch exhibits relatively slow convergence and slightly higher errors, whilst with 1000 batches stronger error fluctuations with increase computational cost was observed. A value of 5 is chosen as good compromise.

$x \in \mathcal{X}$	$\forall \sigma \in \mathcal{Y}$	MRE	
		$U^{235}_{f,2}$	$\Sigma_{a,1}$
-	$\sigma/max(\sigma)$	$3E-1$	$2E-1$
$\sqrt{Bu}$	$\sigma/max(\sigma)$	$2E-1$	$2E-1$
$\sqrt{Bu}$	$\sigma - mean(\sigma) \leftarrow log(\sigma)/max(log(\sigma))$	$1E-1$	$2E-1$
$\sqrt{Bu}$	$\sigma/var(\sigma) \leftarrow \sigma - mean(\sigma)$	$2E-2$	$3E-2$

Table 1: Data normalizations.

N	MRE	
	$U^{235}_{f,2}$	$\Sigma_{a,1}$
2	$2E-1$	$5E-1$
5	$3E-2$	$2E-1$
8	$2E-2$	$4E-2$
15	$2E-2$	$3E-2$
20	$2E-2$	$3E-2$
30	$2E-2$	$3E-2$

Table 2: Increasing N in Shallow ANN.

$f(x)$		MRE	
		$U^{235}_{f,2}$	$\Sigma_{a,1}$
HardS	(see [2])	$2E-1$	$3E-0$
ReLu	$max(0,x)$	$2E-2$	$1E-1$
Elu	$max(0,x) + min(0, e^x - 1)$	$2E-2$	$6E-2$
Sigmoid	$(1+e^{-x})^{-1}$	$2E-2$	$6E-2$
Tanh	$(e^x - e^{-x})(e^x + e^{-x})^{-1}$	$2E-2$	$3E-2$

Table 3: Activation functions.

$\mathcal{L}(\hat{\sigma}, \sigma)$	MRE	
	$U^{235}_{f,2}$	$\Sigma_{a,1}$
$mean(abs(\hat{\sigma} - \sigma))$	$2E-2$	$3E-2$
SmoothL1Loss	$2E-2$	$5E-2$
$mean((\hat{\sigma} - \sigma)^2)$	$2E-2$	$8E-2$
$max(abs(\hat{\sigma} - \sigma))$	$1E-1$	$1E-1$

Table 4: Loss function.

ANN ARCHITECTURE

MRE errors for Shallow ANN with varying number of neurons are presented in Table 2 were $N = 8$ suffices to reach the target accuracy. Actually, error profiles during training are quite similar for $N \geq 8$ at least up to 1E5 epochs. The amount of neurons defines the *degrees of freedom* of the model thus shaping its approximation power, provided a proper training. Fewer neurons imply a simpler parameter space to explore, faster training times and a resulting smaller library size which is also faster to process. We identify a Shallow ANN with $N = 8$ as the smallest network to analyze with the industry data set.

For a fully connected ANN with N neurons, L layers, $I = 3$ inputs and $O = 1$ outputs the amount of parameters $|ANN| = (L-1)N^2 + (I + L + O)N + O$. We compare Shallow and Deep architectures in the search of minimal error and convergence trends using a constant library size. We first consider simple "rectangular" ANN of approximately constant number of parameters layers $L = 1, 2, 3, 4, 5$ and neurons $N = 50, 13, 9, 7, 6$, respectively. The MRE errors for the U^{235}_{f2} are presented in Fig. 1.a where each layer is separated by "/" in the scheme. Errors decrease monotonically up to 1E3 epochs. Then, fluctuations are noticed since the Adam optimizer explores the parameter space to escape a possible local minimum. More than a single layer provides smaller errors and faster convergence, specially for 2 or 3 layers. Test and training errors are similar since cross sections tend to be smooth noise-free functions and training data abundant (with respect to cross section variance) in this 80/20 split. In Fig. 1b we analyze other types of topologies suggested from literature, without noticing further improvement in these settings. The other cross sections show similar results. The 2-layer ANN $13/13$ shows the best performance.

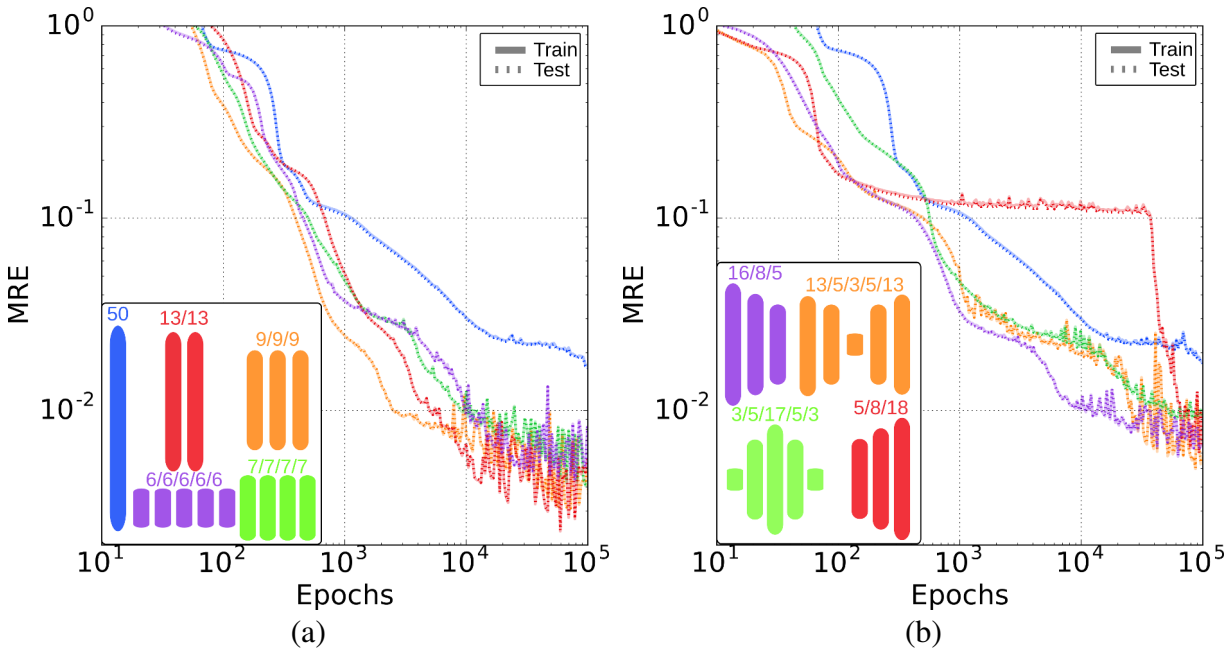


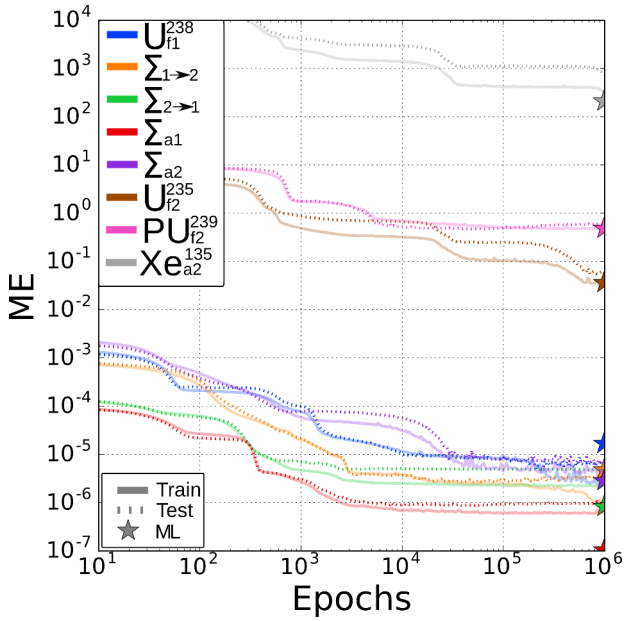
Figure 1: MRE for U_{f2}^{235} for ANN (a) constant number of neurons per layer (b) common architectures with varying neurons per layer.

INDUSTRY DATA SET

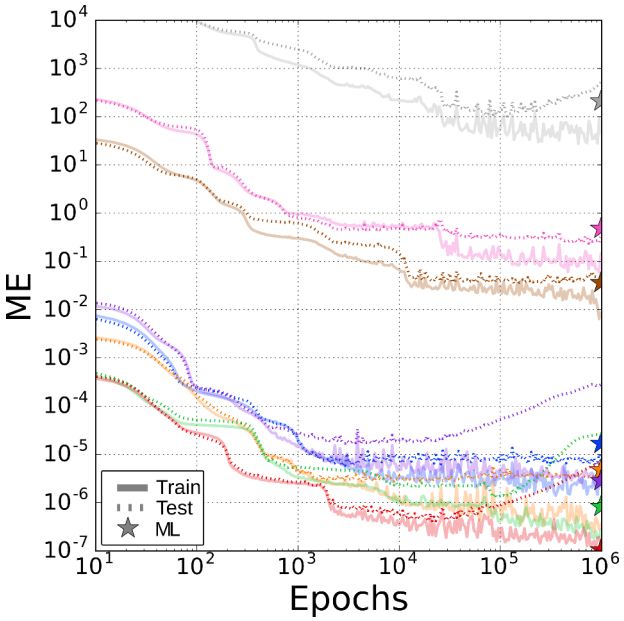
We analyze the two ANN of interest on a smaller discretization of $[35 \times 6 \times 6 = 1260]$ as it could be used in industry for this range of variables. Only 3% of the data is used for training while the rest is for testing. Cross section's ME for the $N = 8$ ANN is presented in Fig. 2.a. Though some separation between test and train error can now be noticed, the network was able to properly generalize even for this significantly smaller data set. Multi-linear (ML) errors marked with a star were in general achieved only after a relatively high amount of epochs. For arriving TE $= 1\text{E}-1$ however, only 6E3 epochs were actually required. ME for the ANN $N = 13/13$ is shown in Fig. 2.b for which a TE $= 1\text{E}-1$ is reached at an even lower number of 7E2 epochs. For this $N = 13/13$ ANN and smaller data set after 1E5 epochs over-fitting starts to occur and test errors stagnate or even increases for some cross sections. Train errors on the contrary keep diminishing even reaching ML for all cross section as the set is being memorized by the network. Different regularization techniques are available for ANN that could be implemented to improve these result such as dropout, batch normalization or a standard L_2 regularization.

An error histogram for the ANN $N = 13/13$ discriminated by cross section is presented in Fig. 2.c at 7E2 epoch. It shows centered means, and low overall errors without tails. In Fig 2.d the TE and the reduction, defined as $100(1 - |ANN|/|ML|)$ i.e., the ratio of parameters in the ANN model with respect to mulit-linear, is presented for these cases. Errors are reported at 1E6 epochs except for the $N = 13/13$ ANN with the industry data set which is at 2E4 epochs (early stop criterion). All ANN errors are lower than $1\text{E}-2$ and a significant reduction of up to 99.7% is obtained. In general, cross section can be considered smooth noiseless functions and if enough train data is available (with respect to the degrees of freedom of the ANN) as is the case with the 80/20 split, train errors are quite representative of test error. ANN are regression approximations

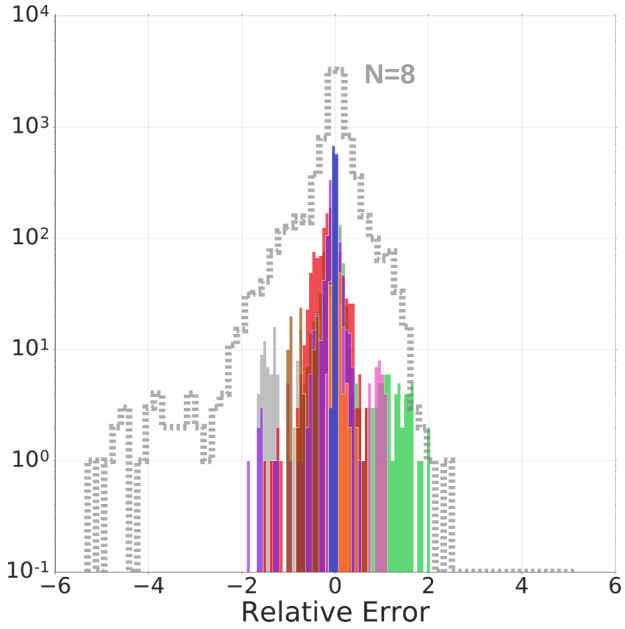
and approaching the accuracies of interpolating models can prove challenging as evidenced by the large amount of epochs required or the over-fitting difficulties with a slightly bigger $N = 13/13$ ANN on a small data set. However other techniques, not explored in this work, allow to further increase the accuracy by adding special ANN that explicitly model the residual of the train error and to augment the reduction in the library size by eliminating weights and biases unimportant in the network.



(a) $N = 8$ (single layer) in industry set.



(b) $N = 13/13$ (two layers) in industry set.



(c)

Model	TE	80/20	Industry
Multi-Linear	Test	1.2E − 3	1.9E − 2
	Size	36096	1260
ANN ($N = 8$)	Train	4.7E − 2	3.9E − 2
	Test	4.4E − 2	7.3E − 2
	$\|ANN\|$	41	
Reduction		99.8%	96.7%
ANN ($N = 13/13$)	Train	9.0E − 3	2.1E − 2
	Test	8.5E − 3	6.0E − 2
	$\|ANN\|$	222	
Reduction		99.4%	82.3%

(d) $N = 13/13$ in industry set

Figure 2: (c) Error histogram discriminated by cross section. (d) Total error (TE) and reduction for the 80/20 % and the Industry data set. Common legend for the cross sections.

5 PERFORMANCE REMARKS

For the ANN considered in this work training and evaluation times are presented in Table 5. The hardware used was a CPU of 24 cores of 2300 MHz and 126 GB of RAM with a GeForce GTX 1080, Python 2.7 and Cuda 10.1 [14]. Training times are given per epoch including a single network adaptation (only one batch equal to the train set). The training time of an epoch ranges from $1\mathrm{E}-3$ to $1\mathrm{E}-1$ depending on the availability of GPU acceleration.

As considered in this work $1\mathrm{E}6$ epochs and 5 batches a non-negligible calculation time of about 3h/ANN/σ may be required whereas for $7\mathrm{E}2$ epochs only a few seconds. An evaluation vector of size $5\mathrm{E}5$ was considered and times are expressed per evaluation point. These were quite compatible with industry requirements about $1\mathrm{E}-5$ seconds especially if GPU acceleration is available.

N	GPU			CPU		
	Training [s/epoch]		Evaluation	Training [s/epoch]		Evaluation
	80/20	Industry	[s/point]	80/20	Industry	[s/point]
8	$2\mathrm{E}-3$	$2\mathrm{E}-3$	$1\mathrm{E}-9$	$7\mathrm{E}-3$	$1\mathrm{E}-3$	$2\mathrm{E}-8$
50	$2\mathrm{E}-3$	$2\mathrm{E}-3$	$2\mathrm{E}-9$	$2\mathrm{E}-2$	$2\mathrm{E}-3$	$3\mathrm{E}-7$
200	$3\mathrm{E}-3$	$2\mathrm{E}-3$	$4\mathrm{E}-9$	$1\mathrm{E}-1$	$5\mathrm{E}-3$	$1\mathrm{E}-6$
$13/13$	$3\mathrm{E}-3$	$3\mathrm{E}-3$	$8\mathrm{E}-10$	$2\mathrm{E}-2$	$3\mathrm{E}-3$	$1\mathrm{E}-7$

Table 5: Training and evaluation times for a single ANN.

6 CONCLUSION

ANN were used to approximate few-group homogenized cross sections for a standard PWR UO2 fuel assembly using a randomly sampled Train/Test split of 80/20 % and a representative industry case. Sensitivity studies were performed for the main aspects of ANN modeling: data normalization, network architecture and training hyper-parameters. Results were compared to multi-linear interpolation in a Cartesian grid, pointing out the capabilities of ANN to reduce the storage requirements for the fitting coefficients.

About the preprocessing, considering the cross section's variance and, to a limited extent, the square root of the burnup showed the best results. Naturally an L_1 loss function outperformed others with respect to the mean absolute error. Changing the number of batches, parameter initialization and activation function showed negligible changes in the error evolution. Values suggested by the literature proved to be adequate, especially for the learning rate.

Provided a constant amount of degrees of freedom (library size), considering 2-3 layers improved the error convergence rate with respect to a Shallow ANN. As evidenced by fluctuation in the training error at a high amount of epochs, a few layers seem to provide a richer parameter space for the optimizer. Architectures with different numbers of neurons per layer did not offer any relevant advantage. With the 80/20 % split, where the train data has a size of $\sim 3.6\mathrm{E}4$, cross sections train errors were systematically representative of test error.

Two ANN models of interest were retained for the industry data set: the first characterized by fast convergence and low error, composed of two layers ($N = 13/13$), and the second one for maximal

reduction of a single layer ($N = 8$). They were able to achieve an acceptable error 0.1% after short training of 6E3 and 7E2 epochs for the ANN of $N = 8$ and $N = 13/13$ respectably. Nonetheless the multi-linear errors were not reached for all cross section even after 1E6 epochs and the $N = 13/13$ showed over-fitting in this smaller industry data set. Evaluation times were quite compatible with the expected industry standard though CPU training times could become prohibitively large, specially for ANN counting more than a few neurons without GPU acceleration. For the two ANN of interest with the two considered train set a significant reduction of up to 99.8% was obtained.

REFERENCES

[1] A. Barreau. "Burn-up Credit Criticality Benchmark." Technical Report Phase, IID, OECD-NEA (2006).

[2] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. "Automatic differentiation in PyTorch." *NIPS* (2017).

[3] R. S. at Al. "APOLLO2 Year 2010." *Nucl Eng and Technology*, **volume 42**, pp. 474–499 (2010).

[4] U. Grundmann et al. "DYN3D version 3.2 - code for calculation of transients in light water reactors (LWR) with hexagonal or quadratic fuel elements - description of models and methods -." *Wissenschaftlich-Technische Berichte* (2019).

[5] R. Zivanovic and P. Bokov. "Cross-section parameterization of the pebble bed modular reactor using the dimension-wise expansion model." *Annals of Nuclear Energy*, **volume 37**, pp. 1763–1773 (2010).

[6] D. Botes and P. M. Bokov. "Polynomial interpolation of few-group neutron cross sections on sparse grids." *Annals of Nuclear Energy*, **volume 64**, pp. 156 – 168 (2014).

[7] J. Dufek. "Building the nodal nuclear data dependences in a many-dimensional state-variable space." *Annals of Nuclear Energy*, **volume 38**(7), pp. 1569 – 1577 (2011).

[8] E. Szames, K. Ammar, D. Tomatis, and J. Martinez. "Few-group cross sections library by active learning with spline kernels." *Proceding of the conference: Physics of reactor conference, Cambridge, United Kingdom* (2019).

[9] H. L. Thi et al. "Use cases of Tucker decomposition method for reconstruction of neutron macroscopic cross-sections." *Annals of Nuclear Energy*, **volume 109**, pp. 284 – 297 (2017).

[10] K. Hornik. "Approximation capabilities of multilayer feedforward networks." *Neural Networks*, **volume 4**(2), pp. 251 – 257 (1991).

[11] S. Mirvakili, F. Faghihi, and H. Khalafi. "Developing a computational tool for predicting physical parameters of a typical VVER-1000 core based on artificial neural network." *Annals of Nuclear Energy*, **volume 50**, pp. 82 – 93 (2012).

[12] B. Leniau and et al. "A neural network approach for burn-up calculation and its application to the dynamic fuel cycle code CLASS." *Annals of Nuclear Energy*, **volume 81**, pp. 125 – 133 (2015).

[13] E. Szames. "FFANN for regression." (2019). URL https://github.com/BetanSz/FFANN-for-scalar-regression.

[14] J. Nickolls, I. Buck, M. Garland, and K. Skadron. "Scalable Parallel Programming with CUDA." *Queue*, **volume 6**(2), p. 4053 (2008).