Neutron spectra reconstruction based on an artificial neural network trained with a large built dataset

Maha Bouhadida, Mariya Brovenchenko, Thibaut Vinchon, Wilfried Monange, Francois Trompier

Neutron spectra reconstruction based on an artificial neural network trained with a large built dataset

Bouhadida M.*, Brovenchenko M.*, Vinchon T.†, Monange W.*, Trompier F.*

*Institut de Radioprotection et de Sûreté Nucléaire (IRSN), Fontenay-aux-Roses 92260, France
maha.bouhadida@irsn.fr, mariya.brovchenko@irsn.fr, francois.trompier@irsn.fr, wilfried.monange@irsn.fr

† Institut de Radioprotection et de Sûreté Nucléaire (IRSN), Cadarache 13115, France
thibaut.vinchon@irsn.fr

INTRODUCTION

Neutron spectrometry is of great significance in different fields as reactors design, nuclear safety and radiation protection. However, determining neutron spectra is a heavy task due to the complexity of neutron interactions and the wide range of neutron energy [1]. Moreover, there is no direct measurement method or neutron fluence energy distribution covering the whole range of energy of neutrons. Bonner spheres spectroscopy (BSS) and activation methods remain the most used approaches for providing accurate determination of the neutron spectrum, but, the measured data needs to be analyzed with suitable spectrum unfolding program. There are several unfolding algorithms to reconstruct neutron spectra. We can cite algorithms based on iteration, maximum entropy, genetic algorithms etc… These approaches have limitations especially the requirement of a prior spectrum [2]. To overcome this, new unfolding methods based on artificial neural networks ANNs has become of interest and different techniques were proposed in recent years to solve the related unfolding problems [3,4,5]. Neural network models [6] are algorithms based on concepts derived from research on the nature of the brain. Contrary to the other unfolding methods needing prior information about spectra, the neural networks explore a training process, which rules out the requirement of pre-known data. However, the proper use of ANN requires the availability of a sufficient size training dataset and an optimization of the data processing (scaling, feature engineering, normalization…). From the recent literature, it is difficult to evaluate the pertinence and the efficiency of ANN compared to other methods of neutron unfolding literature, since the two aspects mentioned above are insufficiently addressed [7].

In our project, we aim to evaluate properly the performance of ANN for neurons spectra unfolding and to compare with other unfolding methods. We also want to go further by combining them in order to create a robust and a reliable solution. In this paper, we present the first step towards our goals which is the dataset building technique allowing the generation of a large number of physical neutron spectra. We also detail the dataset processing and we describe our first implemented architecture for the spectra unfolding.

DATASET BUILDING

As explained above, the major inconvenient of the ANN unfolding techniques is the necessity of a large dataset for the initial training ANN. We propose a database building technique enabling us to generate as many samples as needed in order to achieve a high spectra prediction accuracy. What makes this step original is the conception of an infinite number of "physical" neutron spectra by modeling a simple context (neutron source, geometry and detector). Our database contains initially 16 000 samples. Each sample is a neutron spectrum generated via a Monte-Carlo based SERPENT software [8]. The geometry designed by is composed of 3 concentric spherical surfaces. In other terms, it is about a central sphere s_1 of a radius R_1 and two spherical "shells" s_2 and s_3 surrounding it and having respective radii R_2 and R_3. R_1, R_2 and R_3 are limited respectively by 10 cm, 20 cm and 30 cm. Each cell of this geometry (s_1, s_2 and s_3) is made up by mixing 16 basic materials including for example moderators, vacuum and the air etc. The fractions of the materials have a sum of 1 and are chosen randomly following a continuous probability distribution beta covering the interval [0,1]. Then, the resulting spectra are sufficiently various covering thermal, epithermal and fast neutron cases but still have a "physical" shape.

The neutrons source is also randomly chosen from the fission neutron distribution of the following isotopes list: U-235, U-238, Pu-239, Pu-241. We socre 1001 energy bins between $1e^{-11}$ MeV and 20 MeV for the neutron fluence. The neutrons pass through the 3 cells and their fluence is detected at the third one s_3. A python script allows us to extract this default fluence and its statistical uncertainty from the SERPENT software simulation output.

The spectra set with the shape of (rows = number of samples, columns = 1001) corresponds to the neural network output. The input is, instead, the reactions rates of the detector with the shape of (rows = number of samples, columns = 8). Thanks to our database building technique, we are able to

select different types of detectors but we decide to initially consider the multiple foils neutron activation spectrometer (SNAC2) described in [9]. SNAC2 is dispatched in French Nuclear facility for criticality accident dosimetry purposes. It is a multi-activation foil spectrometer used for criticality dosimetry in case of a criticality accident. It is composed of copper, gold, nickel, magnesium and indium foils sealed in a cadmium box of 22 mm diameter. It also has two Cu foils outside of the described case (front and back). The reaction rates are then computed following Equation (1) where φ_i is the fluence at the i^{th} bin with i=1,..,n=1001, RF_{ji} is the j^{th} response function value at the i^{th} bin with j=1,..,8 and R_j is the jth reaction rate. In fact, the response functions are "pre-calculated" by SERPENT simulation using a detailed SNAC2 activation foils modeling.

As you can conclude, we can generate the required number of samples while imitating "realistic" physical constraints to efficiently train our neural network.

$$R_j = \sum_{i=1}^{n} RF_{ji} * \varphi_i \quad (1)$$

We plot in Fig. 1. a couple of (reactions rates, spectrum) included in our dataset.

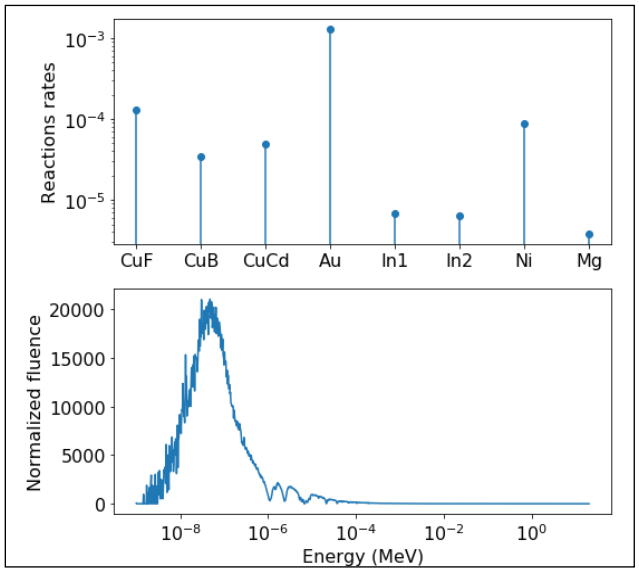

Fig. 1. Visualization of a couple (reactions rates, spectrum) included in our dataset. The normalized fluence corresponds to the fluence divided by the energy interval and the x-axis is "log" scaled.

DATASET PROCESSING

Data processing is a mandatory step to succeed the ANN model training step. However, it has not been of a huge interest in the recent papers. We propose a feature engineering, a data scaling and a data augmentation to prepare our dataset to training. The feature engineering is applied to the reactions rates. It is about computing the ratios between all the reactions rates pairs and adding the computed vectors to the ANN model input. Then, the input shape becomes (number of samples, 64=8 + all computed ratios). After that, this input is scaled. The two most popular techniques for scaling data prior to modeling are normalization and standardization. We choose the standardization which scales each input variable separately by subtracting the mean (called centering) and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one. We have implemented a data generator allowing the application of data augmentation to the spectra and reactions rates. For each epoch, we randomly draw a permutation of the spectra and associated reactions, to always read the data in a different order following the principle of stochastic gradient descent. For the output (the spectra), we add a padding (zeros at the beginning of the vectors) to have a shape of (numbers of samples, 1024) instead of (numbers of samples, 1001). This new shape is more efficient to build the ANN model.

PROPOSED ANN MODEL

As explained before, ANNs are a set of input nodes that link directly or indirectly (through hidden layers) to a series of output nodes. A weight is associated with each link between the nodes. During a training process, the ANN model tries to acquire the rules producing the outputs from the inputs and then, the weights are adjusted to present the closest outputs to the real ones. A standard network structure with several layers is the multilayer perceptron MLP. It is a fully connected and feedforward model.

To mathematically describe the considered MLP model, we denote respectively the input of 64 elements and the output of 1024 elements by X and Y. We fix h_0=X and we compute, for each layer l:

$$h_l = DenseNorm(h_{l-1}) \quad (2)$$

where l = 1, . . . , L with L is the total number of layers. Then, Y = h_l. All DenseNorm layers have the following structure:

$$DenseNorm(h) = Act(Norm(W \times h + b)) \quad (3)$$

where W and b are the trained parameters. Norm is a batch or a layer normalization if needed and Act is the activation function. A dropout is applied if needed. We can note such an architecture:

$$Y = MLP(X) \quad (4)$$

We propose a "two-branch" MLP model where we add a second "branch" corresponding to a separate MLP, with a last layer that contains a single neuron (so one number per

spectrum) used to normalize it. Our model can be written in the way below:

$$\begin{cases} Y_1 = MLP_1(X) \\ s = MLP_2(X) \\ Y = Y_1 \times s \end{cases} \quad (5)$$

Therefore, each spectrum is constructed as the product between the output (1024 bins) of MLP_1 and the output of MLP_2. The normalization branch allows us to have better metrics compared to the standard MLP structure.

The first model that we are actually training contains 5 layers with 64, 128, 256, 512, 1024 neurons for the "main" branch (MLP_1). It evolved 4 layers with 64, 32, 8, 1 neuron(s) for the normalization branch (MLP_2). The rectified linear activation function ReLU is applied to all layers and no dropout is considered. It is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance. is

In these studies, we use python scientific libraries for the dataset processing and Tensorflow libraries for the training and evaluation processes. We use the callback TensorBoard to track the training in real time, and we use the mlflow library to save easily all our experiments results.

RESULTS

As we use non-normalized spectra, which can take values quite large at certain energy bins, we found that the quadratic loss often leads usually to overflows (numerical instability). We therefore preferred to use the mean absolute error MAE as a loss function to compare the results of our experiments on validation samples. For all our experiments, we use the Adam optimizer [10] which is today one of the most used in deep learning because it leads to more stable and simple training adjustment. We use a learning rate of 0.001 and consider 75% of the dataset for training and the other 25% for validation.

Our model is trained with early-stopping (using the callback EarlyStopping) with a patience parameter equal to 200 and an epoch maximum number of 5000, which corresponds to a very large epoch number. The principle of early-stopping is as follows: if there is no improvement in the validation MAE during patience=200 epochs, we stop the training and we reload in memory the parameterization of the network observed in previous epochs with the best validation error. We note that the training is interrupted in our experiments before reaching the maximal number of epochs by early-stopping. We use in all our experiments mini-batches of size 512, to obtain results faster than with smaller mini-batches, even if the use of mini-batches of the order of 32 or 64 often lead to better results on the validation error.

As a first result, a validation MAE of 14.89 is obtained at the 401st epoch considered as the restored early stopping epoch for a total epoch number equal to 601. The training duration is about 1.7 mn since it was performed on an Nvidia Tesla V100 GPU (32GB of GPU memory).

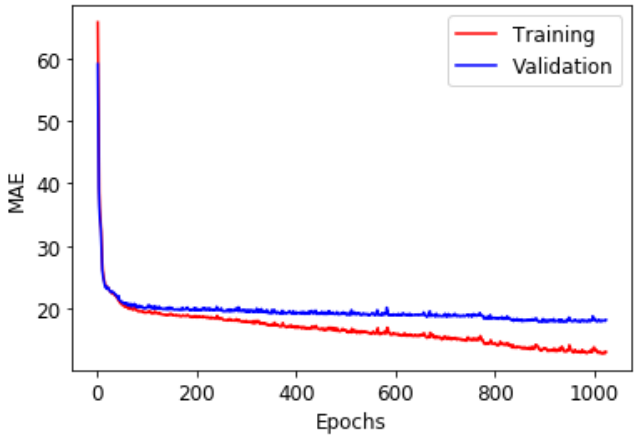

Fig. 2. Training and validation MAE plots as a function of the epochs

The convergence curve (training and validation MAE) observed during the epochs is displayed in Fig. 2. and an example of spectra reconstruction is proposed in Fig. 3.

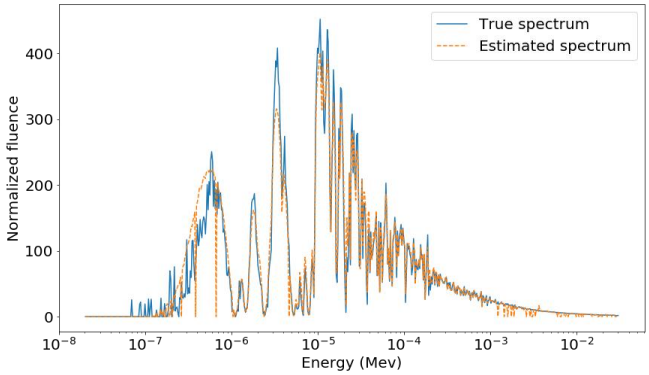

Fig. 3. Example of an estimated spectrum using our model. The normalized fluence corresponds to the fluence divided by the energy interval and the x-axis is "log" scaled.

Despite the fact that we do not use batch/ layer normalization or dropout, we obtain performances without strong overfitting (as observed on the training curve in Fig. 2). But, it will be necessary to try to hyper-optimize the dropout and normalization parameters to achieve higher performances and that will be the next step of our project using the optuna library [11] which is the most efficient nowadays.

CONCLUSION

In this paper, we detailed the dataset conception and its processing. Our dataset can include an infinite number of simulated physical neutron spectra based on a simple context. We also described the first architecture implementation and discussed the results. The normalized version of MLP has better performances than the basic MLP version for the same parameters. We insist that neural networks can be very efficient in predicting spectra which makes our model be a promising candidate for a reliable unfolding. Future considered steps are to go further by adding more metrics and testing different optimizers. Optimizing the hyper-parameters remain the must-to-do task. We will also implement and test other models based on convolutional transpose layers and on residual layers. Then, we will compare the neural network efficiency with the classical methods one and we will even try to combine then in order to build a hybrid method. Finally, we will expand the applicability of our model for different fields to obtain the absorbed dose and kerma in tissue values in addition to the spectra.

REFERENCES

1. J.M. Ortiz-Rodirguez et al., "A neutron spectrum unfolding computer code based on artificial neural networks", Radiat. Phys. Chem., 95 (4) 428-431 (2014).
2. M. Reginatto, "Bayesian approach for quantifying the uncertainty of neutron doses derived from spectrometric measurements", Radiation Protection Dosimetry, 121, 1,64-69 (2006).
3. S.A. Hosseini et al., "Neutron spectrum unfolding using artificial neural network and modified least square method", Radiation Physics and Chemistry, 126, 75-84 (2016)
4. J. Wang et al., "Neutron spectrum unfolding based on generalized regression neural networks for neutron fluence and neutron ambient dose equivalent estimations", Applied Radiation and Isotopes, 154, 108856 (2019).
5. C. Cao et al., "An adaptative deviation-resistant neutron spectrum unfolding method based on transfer learning", Nuclear Engineering and Technology, 52, 2452-2459 (2020).
6. Vega-Carillo et al., "Artificial neural networks technology for neutron spectrometry and dosimetry", Radiation Protection Dosimetry, 126 (1-4), 408-412 (2007).
7. N. Mohammadi et al., "Neural network unfoldingof neutron spectrum measured by gold foil-based Bonner sphere, J. Radioanal. Nuc. Chem, 303 (3), 1687-1693 (2015).
8. J. Leppanen, "Serpent – a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code", User's manual (2015).
9. F. Trompier et al., "Dosimetry of the mixed field irradiation facility CALIBAN", Radiation Measurements, 43, 1077 – 1080 (2008).
10. D. P. Kingma et al., "Adam: A method for stochastic optimization. arXiv", preprint arXiv, 1412.6980 (2014).
11. T. Akiba et al., "Optuna: A next-generation hyperparameter optimization framework.", Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2623–2631 (2019).