BJRS

BRAZILIAN JOURNAL
OF
RADIATION SCIENCES
11-01A (2023) | 01-20 | e2201

Modeling of the mass attenuation coefficients of X ray beams using deep neural networks (DNN) and NIST database

Silva[a] G.B., Botelho[a] V.R., Becker[a] C.D., Viccari[b] C., Pianoschi[a] T.A.

[a] *Universidade Federal de Ciências da Saúde de Porto Alegre, Porto Alegre, Rio Grande do Sul, Brazil*

[b] *Faculdade de Filosofia, Ciências e Letras (FFCLRP) da Universidade de São Paulo, Av. Bandeirantes, 3900 - Vila Monte Alegre, Ribeirão Preto - SP.*

gustavobs@ufcspa.edu.br

ABSTRACT

Attenuation coefficients are essential physical parameters for many applications, such as the calculation of photon penetration and energy deposition to evaluate biological shielding. Estimating these parameters is complex, making it necessary to apply more sophisticated methodologies. The objective of the present study was to propose a model for estimating the attenuation coefficients using artificial neural networks. The NIST database was used to estimate the attenuation coefficients in terms of energy and atomic number from a regression problem using two approaches: the proposition of an automated model using the framework Talos and a manual model using Keras. The characteristics of the best model proposed in Talos were applied in manual training via Keras with cross-validation to evaluate the learning curves. The following were also assessed: the comparison of the curves of the attenuation coefficients predicted by the model compared with the reference data and the general comparison of the vectors X and y of the two models discussed. The Talos framework reference model obtained the following values of Loss and MSE error metric: 0.13 and 0.037, respectively. The best model of the manual approach received the following results: 0.19 and 0.08 for the loss function and MSE error metric, respectively. The absolute percentage error (MAE) of the difference in the results between the two models was: 0.065 and 0.044 for the Loss and MSE metrics. Despite applying two distinct propositions, both models had the same difficulties in predicting discontinuities in the physical behavior associated with the attenuation coefficients.

Keywords: **Mass attenuation coefficients, neural networks, deep learning.**

ISSN: 2319-0612
DOI: https://doi.org/10.15392/2319-0612.2023.2210
Submitted: 2022-11-30
Accepted: 2023-11-16

1. INTRODUCTION

Since the discovery of X-rays to the present day, numerous applications associated with this type of radiation have been evaluated and improved. In the context of using X-rays as a diagnostic and medical treatment tool, it was found that the number of photons absorbed and transmitted from the interaction of X-ray with human tissue can be theoretically evaluated from the analysis of two key attenuation coefficients, namely the mass attenuation coefficient (μ/ρ) and the mass absorption coefficient $(\mu en/\rho)$ [1]. These coefficients are crucial for characterizing the penetration of radiation into a medium and understanding how X-rays interact with matter [2].

Hubbell (1982) established a formalism for defining the attenuation coefficients based on the Lambert-Beer Law. According to the author, a narrow, monoenergetic beam of photons is attenuated to intensity I from an incident intensity I_0, passing through the thickness of material x, following an exponential decay [3].

The first comprehensive analysis of mass attenuation coefficients was provided in Report 33 by the International Commission on Radiation Units and Measurements (ICRU). According to ICRU Report 33, the mass attenuation coefficient is defined as the ratio of the linear attenuation coefficient to the material density. It is expressed in units of cm^2/g and quantifies the probability of attenuation per unit length traveled by ionizing radiation within a given material [4]. In the realm of estimating mass attenuation coefficients, various models have been explored in the scientific literature [1-8]. These models typically employ a semi-empirical approach, combining experimental and theoretical insights to overcome the limitations of purely experimental models. The challenge lies in estimating attenuation coefficients across a broad spectrum of energy levels, given the diverse range of materials utilized in practical applications [9]. Other approaches have also proposed models to determine the attenuation coefficients using Monte Carlo Simulations, which sometimes required extensive computational processing times [10-11].

Despite the diversity of methodologies used to derive attenuation coefficients, the models established by the National Institute of Standards and Technology (NIST) have emerged as standard references [9-16]. NIST's models are acclaimed for their sophisticated theoretical calculations, providing estimates for different energy intervals [4, 17]. However, they come with inherent limitations. Notably, NIST data offer coefficients at discrete data points, resulting in substantial intervals between these points. Consequently, some applications necessitate more explicit models, such as the indirect method for estimating spectra, which entails calculating the derivative of μ [18].

An illustrative example of an alternative model can be found in the work of Manjunatha *et al.* (2017), designed to estimate mass attenuation coefficients across a wide energy range spanning from 1 keV to 20 MeV. The approach involves fitting empirical equations that establish a robust relationship between mass attenuation coefficients and energy, encompassing elements from Z = 1 to Z = 92. To enhance precision, the authors further segment the energy range into sub-regions, meticulously selecting polynomial functions that deliver the optimal fit within each specific region [8].

In the context of machine learning algorithms have proven valuable, especially in cases of nonlinear regressions. Therefore, the analysis of complex physical parameters using machine learning algorithms has gained traction [9]. Consequently, numerous applications to model the physical properties of materials have been proposed in the literature, demonstrating the potential of machine learning techniques in this field [12-15].Machine learning encompasses a wide spectrum of techniques and models, ranging from conventional approaches to the cutting-edge realm of Deep Neural Networks (DNNs).

DNNs, characterized by their intricate multi-layered architecture, represent the pinnacle of machine learning capabilities, enabling the discovery of intricate patterns and relationships within data that were previously challenging to discern. These networks have revolutionized fields such as image recognition, natural language processing, and data analysis, making them a cornerstone of modern AI applications.

Few studies [3,9,16,20] have explored NIST data to develop other applications involving machine learning such as the proposition of artificial neural networks. Furthermore, some of these [9,16] studies used specific energy ranges or estimations of attenuation coefficients for specific materials. Thus, proposing a model that estimates the attenuation coefficients from a DNN that covers all atomic numbers and the energy range proposed by NIST is necessary.

One significant advantage of employing deep neural networks (DNNs), as opposed to traditional Monte Carlo-based models, lies in computational efficiency, as demonstrated by the following example. Consider a Monte Carlo-based code, called TASMIC, designed to calculate photon fluence within the energy range of 20 keV to 640 keV. This conventional approach demands extensive computational resources, typically exceeding 5,000 hours of simulation time to yield results [12]. In contrast, the DNN method, represented here by Talos, accomplishes the same task in a mere 6.5 hours.

This substantial disparity in computational time underscores the efficiency gains associated with the utilization of DNNs. It highlights a pivotal motivation for this study, namely, the pressing need for a more time-efficient approach to estimating attenuation coefficients. Moreover, conducting a comparative analysis between the outcomes generated by this neural network-based method and those from traditional Monte Carlo models can offer valuable insights into both the accuracy and computational expediency of the proposed approach.

This paper aims to explore the data provided by NIST to estimate mass attenuation cofficients $\left(\frac{\mu}{\rho}\right)$ for X-ray beams from a regression problem in a DNN. The research contributes to the broader fields of medical radiology and materials science by potentially improving the accuracy and efficiency of these coefficient estimations through the application of deep learning techniques. The proposed DNN-based method has the potential to reduce computational resources and time while maintaining or enhancing precision. The study comprehensively analyzes various metrics, including loss functions, validation, number of folds, and learning curves, to assess the model's effectiveness. The subsequent sections will provide detailed insights into the methodology, results, and discussions, shedding light on the advantages offered by this approach.

2. MATERIALS AND METHODS

The implementation of the neural network was done in *Python* language, from prototyping via *Keras* and using the framework *Talos* for automatic adjustment of the initial hyperparameters and subsequent manual training to evaluate the training curves. Other libraries such as *Tensor Flow, Scikit Learn*, *Numpy*, and *Matplotlib* were used to analyze and evaluate model metrics.

The methodology of this study involves the implementation of neural networks using two distinct approaches: the utilization of Keras, an API designed for rapid experimentation with deep neural networks, known for its ease of use, modularity, and extensibility; and the Talos framework, which revolutionizes the conventional deep learning workflow by automating hyperparameter tuning and model evaluation [13].

2.1. Pre-processing of the dataset

The definition of the dataset involves using numerical data from NIST tables. These numerical data consist of values of attenuation coefficients in the function of the energy for x-ray beams and the attenuation material (in terms of atomic number Z from Z = 1 to Z = 92). Although the NIST tables contain various parameters, such as mass attenuation coefficients, mass absorption coefficients, parameter Z/A, compounds, and mixtures, conventional DNN training allows the model to output only one parameter at a time. Therefore, mass attenuation coefficients were chosen as the analysis parameter for this study. This choice was motivated because the authors were in search of an alternative model to estimate these mass attenuation coefficients for another application [14] of the indirect method of obtaining the X-ray spectrum [21].

To import the NIST tables, we used the library *Physdata* of *Python*. The *Physdata* consists in a module that imports a package with all the tables of NIST in a simple and rapid format. The data from *Physdata* are organized in a Data Frame from the library Pandas. After importing the tables and adjusting these tables in Data Frame, the Data Set had a total size of 4479 lines, with four columns: Energy (in MeV), mass attenuation coefficients $\left(\frac{\mu}{\rho}\right)$, mass absorption coefficients $\left(\frac{\mu en}{\rho}\right)$ and the atomic number from Z = 1 to Z = 92.

So that the input data for training the model could be created from the defined Data Frame, it was necessary to include the fourth column, related to the elements of atomic number Z. This step was required because there was no available function of *Physdata* that would print the attenuation coefficients for different ranges of atomic number Z. In this way, a *loop* structure was defined that would scan the library function over the entire range of Z and then from the Pandas concatenation function (*pandas.concat)* joins all the values into a single table. The table 1 shows the format of the dataset after the manipulation described above. The dataset had a total size of 4479 lines.

Table 1: The Dataset structure.

Energy (MeV)	Physical parameters		
	$\left(\frac{\mu}{\rho}\right)$ (cm^2/g)	$\left(\frac{\mu en}{\rho}\right)(cm^2/g)$	**Z**
0.0010	7.21700	6.82000	1
0.0015	2.14800	1.75200	1
0.0020	1.05900	0.66430	1
0.0030	0.56120	0.16930	1
0.0040	0.45460	0.06549	1
...	...	...	...
6.0000	0.04583	0.2829	92

After manipulating the dataset from the Data Frame definition, the neural network inputs were defined. Thus, the Energy (MeV) and Z columns (Table 1) were used as input data for the neural network to estimate the mass attenuation coefficients. The last step of processing the data set was to adjust the axes on a logarithmic scale. This step was necessary because the energy ranges of the NIST tables are extensive, and this difference could hurt the regression results.

2.2. Analysis metrics

In this work, two metrics were employed to quantitatively evaluate the results. The first metric used to validate all discrepancy analyses in this study was the Mean Absolute Error (MAE) and the Mean Squared Error (MSE), which are commonly utilized measures to assess the performance of prediction models or data fitting.

MAE is calculated as the mean of the absolute differences between the measured values and the values predicted by the model, while MSE is calculated as the mean of the squared differences between these values. Both MAE and MSE are widely used metrics in deep learning to gauge the accuracy and performance of models.

The mathematical definitions of MAE and MSE are as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}| \tag{1}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \tag{2}$$

Where:

- (n) is the total number of samples.
- (y_i) represents the observed value for sample (i).
- $(\widehat{y_i})$ represents the predicted value by the model for sample (i).

2.3. Adjust of hyperparameters of training with Talos framework

The training method based on Talos involves the performing distinct steps: definition of hyperparameters dictionary; experiment setup in Talos via association with the model in TensorFlow Keras; evaluation of the experiment and choice of the best model to from the analysis of previously selected metrics (*loss* function) and other chosen error metric in regression [24]. In the dictionary definition, the network architecture with the hyperparameters and the number of layers are predicted.

The study encompasses key hyperparameters and parameters essential for deep learning models. Hyperparameters, including the loss function, error metric, activation functions, and optimizers, play pivotal roles in training neural networks. Node parameters, represented by the first to fourth hidden layers, enable the network to capture complex patterns, hierarchical representations, and intricate patterns in the data. The Dropout technique helps prevent overfitting, while batch size impacts model convergence and memory usage. For more in-depth information on these parameters, refer to Chollet *et al.* (2021) [25].

Table 2 shows the selected parameters for the *Talos* experiment. The *Talos* model combines all hyperparameters and hidden layers in the experiment setup. To facilitate the posterior migration of the model to the second training with *TensorFlow* Keras, the input layer and the output layer were selected in fixed form, without the variation of these parameters. In other words, the experiment tested the parameter variation only for the hidden layers. According of Table 2, the error metrics selected for the *loss* function were the MAE and the MSE. For the analysis of error metrics, MSE is selected.

Table 2: Dictionary parameters of Talos experiment

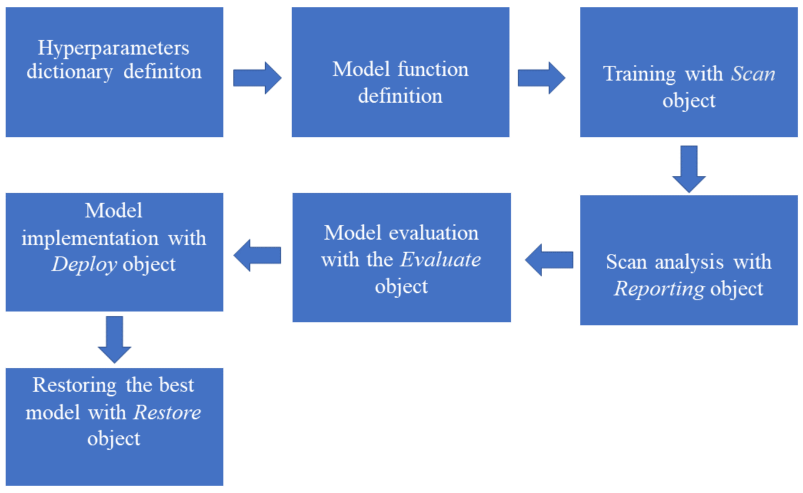
Hiperparameters dictionary	**Values**
Learning rate	0.001
Batchsize	[2000]
Number of epochs	[400]
Loss function	[‘MAE’, ‘MSE’]
Error metric	[‘MSE’]
Activation function 1	[‘relu’, ‘elu’]
Activation function 2	[‘sigmoid’]
Optimizers	[‘ADAM, ‘RMSprop’]
Node parameters	**Attributes**
Layers type	Dense
Input layer nodes	2
Output layer nodes	1
First hidden layers	[4,7,12,24,48,64]
Second hidden layers	[2,4,6]
Thirds hidden layers	[5,6,7]
Fourths hidden layers	[1,4,7]

Talos can be applied with any Keras model without changing the model and without having to structure a new syntax [24]. However, some adjustments need to be made for the case of a model that was previously defined in TensorFlow Keras, involving the following steps:

1. Addition of the input parameters in the dictionary to the model function;
2. Replacement of hyperparameter entries defined in Talos by references to the dictionary of parameters described in table 2;
3. Check if the model is storing the historical object;
4. Modifying the model output.

Figure 1 shows a summary of the implementation steps in Talos. After the dictionary definition and the model function, the experiment is started from the Scan reading object. The input parameters of the digitization project are X and Y parameters of the model input, the name of the model function defined in the previous step, and a name for the experiment. each adjustment attempt of the model in the experiment is called a round and the the number of rounds depends on the number of nodes defined in each attribute of the dictionary and the number of epochs defined.

Figure 1: Implementation steps of model in Talos framework


Hyperparameters dictionary definiton
Model function definition
Training with *Scan* object
Scan analysis with *Reporting* object
Model evaluation with the *Evaluate* object
Model implementation with *Deploy* object
Restoring the best model with *Restore* object

After the Scan step, the experiment results are stored per round without a corresponding log, whose format is a .csv file stored in the job. Thus, to access and evaluate the model, the *Reporting* object is used. Such an object returns a Scan report performed, and the results can be accessed from doing analysis. From *reading*, the experiment's measurements, the model can be evaluated with the *Evaluate*. This *Evaluate* process involves an analysis of the experiment with K-folds cross-validation. Once the correct model is found, the next step is to create a package of deployment with the *Deploy* object. This package can be easily transferred to other environments. The best is easily chosen from the choice of best metric, being the most usual for the problems of measurement is the validation of the chosen error metric is a (*val_metric*).

The *Deploy* package is a compacted file, which contained: the details of the *Scan* object model weights, the model in JSON, the results of the former experiment and samples of the X and y data. To access *Deploy* package data, you need to use the object *Restore*. The *Restore* object consists of the object's assets Scan associated with the experiment, with the chosen model. With this object, the best model can be used to make predictions in any form to use the model objects Keras [21].

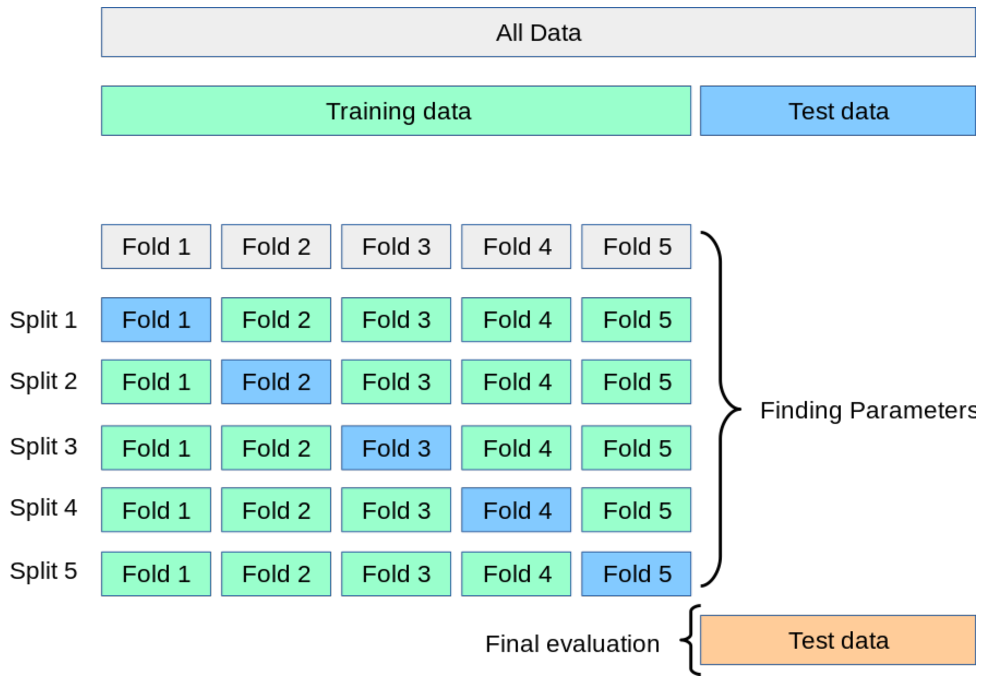
2.4. Manual training with *Keras*

In this step, the architecture, the hyperparameters of the best model proposed in the Talos experiment, and the error metrics that best fit were selected for manual training via Keras. In this approach, the best model is understood to have the lowest rates for the loss function and error metrics. The Talos Reporting object, described in Figure 1, was used to select the information from this reference model.

In this way, the parameters of the Talos reference model (Table 2) were applied in a training manual via Keras with the cross-validation method. Cross-validation is a valuable technique for assessing the performance of machine learning models. It serves the purpose of comparing and selecting the most suitable model for a given predictive task, offering advantages such as lower bias and ease of implementation.

One commonly used variant of cross-validation is k-Fold cross-validation, where 'k' represents the number of partitions into which the dataset is divided. In many cases, 'k' is set to either 5 or 10, but it can vary based on dataset characteristics. In this methodology, the dataset is partitioned into K equal-sized segments. Within each segment (referred to as i), a model is trained on the remaining K – 1 segments, and its performance is assessed on segment i. The final score is computed as the mean of the K scores acquired. This approach proves advantageous when the model's performance exhibits notable variation across different train-test splits. It's important to emphasize that, similar to hold-out validation, the utilization of K-fold cross-validation does not eliminate the necessity for a distinct validation set, which is still required for model calibration. Figure 2 visually outlines the K-fold cross-validation process, and Listing X offers a straightforward implementation example [25].

Figure 2: Cross-validation method



Source: Pedregosa *et al* (2011)

In this particular study, a k value of 10 was employed, meaning the dataset was divided into ten equal parts or folds. For this approach, the database was split into two sets: 75% of the data for training and validation, and the remaining 25% for testing and the final evaluation of the model. The algorithm then trained and validated the model on each of these folds iteratively, ensuring comprehensive evaluation and averaging the results to obtain the final score. This approach helps mitigate issues related to variance in model performance during the train-test split and contributes to a robust assessment of model efficiency.

The analysis of the neural network's results involved evaluating attenuation coefficients for four specific materials: Beryllium (Z=4), Aluminum (Z=13), Copper (Z=29), and Tungsten (Z=74). These materials were chosen based on their relevance in the context of medical radiology, particularly in X-ray applications. The study focused on these specific elements to assess the network's performance in predicting mass attenuation coefficients for materials commonly used in radiological applications.

The results obtained from the neural network were compared with available reference NIST data. This comparison was conducted using a discrepancy analysis to quantify the differences between the neural network predictions and the reference data. The criteria for this comparison included metrics such as Mean Absolute Error (MAE) and Mean Squared Error (MSE) to assess the accuracy and performance of the neural network in predicting mass attenuation coefficients for the selected materials. The goal was to evaluate how well the neural network's predictions aligned with established reference data for these materials commonly used in medical radiology contexts.

Table 3: Talos reference model parameters selected

Hiperparameters dictionary	**Values**
Loss function	MAE
Error metric	MSE
Activation function 1	ELU
Activation function 2	SIGMOID
Optimizers	RMSprop

Node parameters	**Attributes**
First hidden layers	48
Second hidden layers	4
Thirds hidden layers	6
Fourths hidden layers	7
DropOut	0.20
Batch Size	2000
Epochs	400

3. RESULTS AND DISCUSSION

The experiment using Talos (based on the parameters defined in Table 3) lasted about 6.5 hours. Table 4 shows results for the best model metrics obtained with the Talos experiment, that is, the one that obtained the lowest values for the loss metrics (MAE) and the MSE metric.

Table 4: Results of best model in Talos experiment

Metrics	**Values**
Loss function	0.1346
MSE	0.0374
Loss validation	0.1364
MSE validation	0.0414

The results of manual training via Keras for the cross-validation approach, separated per fold are found in table 5, and the learning curves, separated by metric and the number of folds are shown in figure 3.

Table 5: *Loss* function and MSE per fold

Fold Number	***loss* function**	**MSE**
Fold 1	0.203	0.090
Fold 2	0.209	0.087
Fold 3	0.206	0.089
Fold 4	0.199	0.081
Fold 5	0.206	0.086
Fold 6	0.202	0.087
Fold 7	0.203	0.088
Fold 8	0.208	0.085
Fold 9	0.206	0.088
Fold 10	0.213	0.090

Figure 3: Learning curves for: (a) loss (training) vs epochs per fold (b) *loss* (validation) vs epochs per fold (c) MSE (training) vs epochs per fold (d) MSE (validation) vs epochs per fold

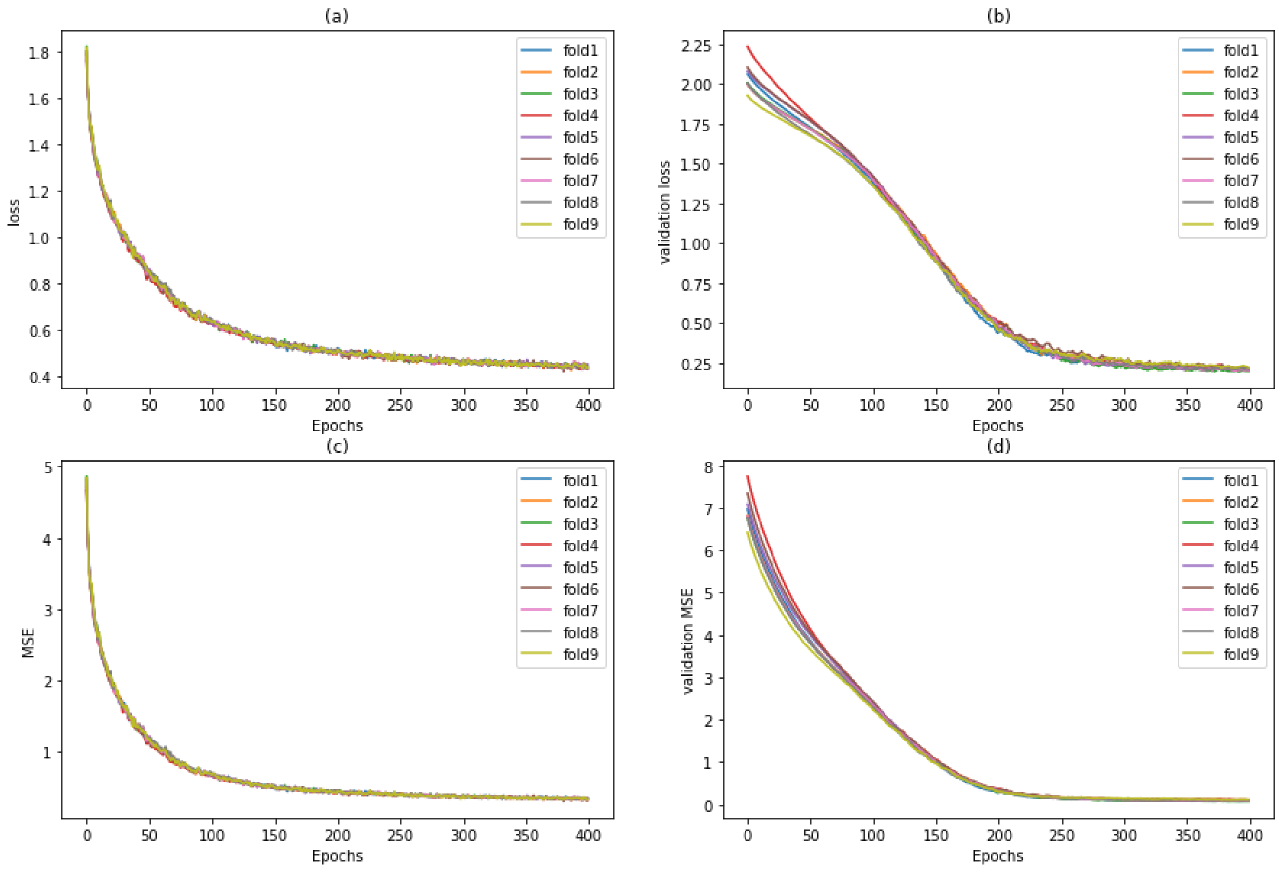


Figure 4 shows the graphic comparison of the physical behavior for the atomic numbers defined in the data in the Talos experiment. And figure 5 shows the general comparison between the X and y vectors of the data measured against the predicted information for the training set and for the test set in the best result model. A visual analysis of figure 5 demonstrates a consistency concerning the highest volume of the training data (75% of the database) compared to the test data (25% of the test data).

Figure 4: Predicted and measured comparison: (a) Beryllium (Z=4) (b) Aluminum (Z=13), (c) Copper (Z=29) (d) Tungsten (Z=74).

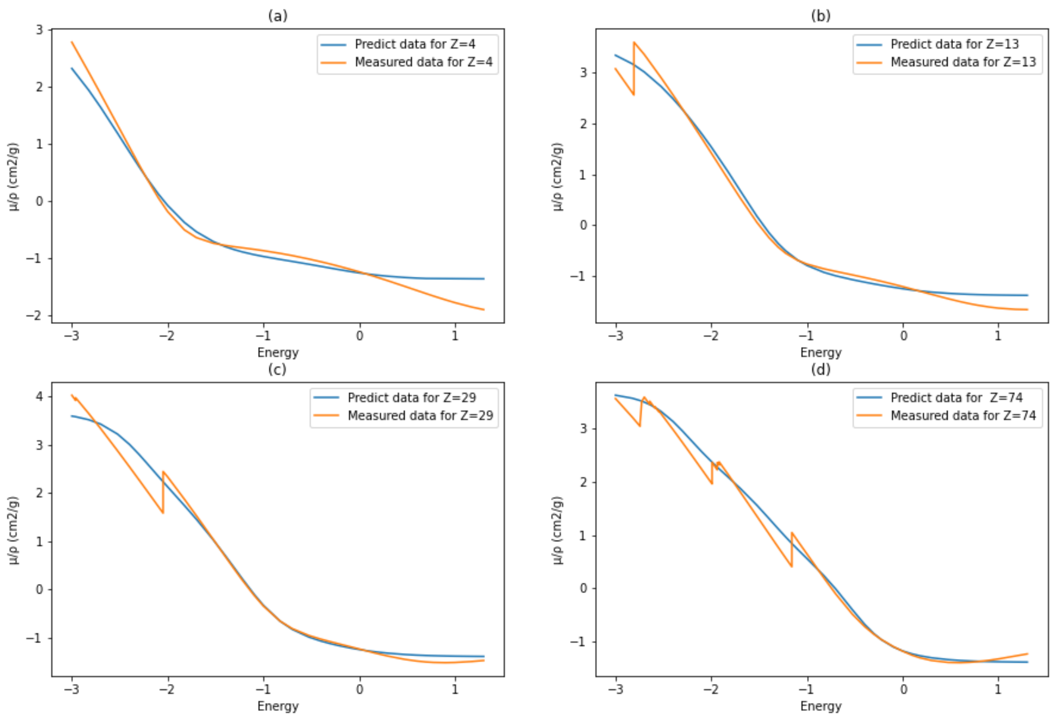
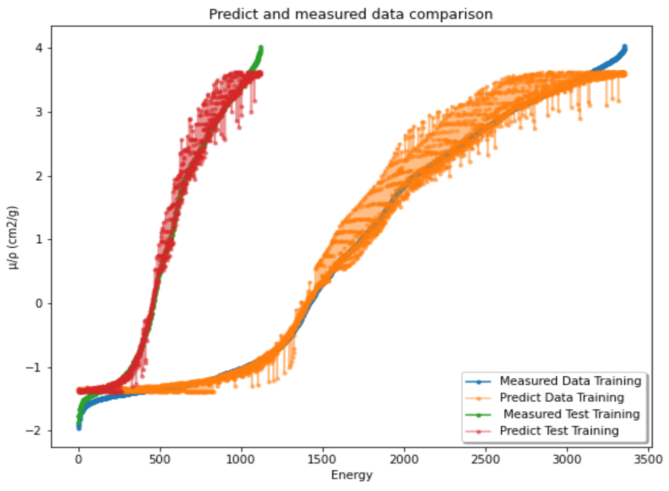


Figure 5: General Predicted and measured comparison.



Regarding the analysis of the models, the best result for training via cross-validation was achieved by fold 4. Examining the table reveals discrepancies between the results of manual training with 10-fold cross-validation and those obtained through training with Talos. A comparison between tables 4 and 5 leads to the conclusion that the Talos-based training approach, without the inclusion of the cross-validation process, yielded the lowest values for the error metric defined for the loss function.

For Loss Metric, the value of MAE is 0.065 and for MSE metric, the value of MAE is 0.044, demonstrating that the most significant absolute average error occurred for the loss function metric (loss). The choice of this error metric to compare the result of the two models was defined in one step after the acquisition of results. This was because it was not expected that the application of the same neural network architecture and exact values of the application's hyperparameters via Talos would change the associated results in the transition of data from one model to the other, proposed in two different codes. Although these codes have been analyzed, an explanation has not been found for these difference results. For future studies, it becomes necessary to understand these differences better so that the methodology proposed in this study is reproducible in other applications.

It is important to clarify that these loss functions and error metrics do not have established reference values, as their primary goal is optimization rather than reaching specific thresholds. Practitioners focus on minimizing loss and improving model performance through techniques such as cross-validation and hyperparameter tuning.

Furthermore, the choice of a manual training approach via Keras was in the fact that despite the framework Talos will provide different attributes for evaluating the action of the models trained in the proposed experiment, it does not allow evaluation of the training curves of these models after the end of the experiment, just assessing the possible rate of learning said during the training [21].

Thus, this condition ends up limiting the analysis these necessary curves since the follow-up to living off of learning rates turns out to be a chore due to the volume of data and the long access time that would need to be done without analysis since training via Talos takes hours to complete accomplished; in the case of the approach of this study, it required a 6.5-hour live follow-up.

A visual analysis of figures 3,4 and 5 shows that the curves tend to have similar physical behavior in predicting mass attenuation coefficients. However, the evaluation of figure 4 demonstrates difficulty in predicting discontinuities associated with the data NIST reference. Even if the best models were selected, the network could not accurately predict these discontinuities.

A possible physical explanation for this behavior may be associated with the data source of the NIST database, which uses a semi-empirical approach, mixing experimental data with theoretical data to evaluate the behavior of radiation in different materials of atomic number. Such behavior, as it is peculiar and challenging to predict, may have caused the neural network to be unable to adjust and model the breaks in the orange curves (Figure 5). Yet another factor associated with the complexity of predicting mass attenuation coefficients may be associated with the fact that the nature of these phenomena is probabilistic and may make it difficult to model these parameters with the proposed dataset alone.

4. CONCLUSIONS

This study successfully introduced a promising model for estimating mass attenuation coefficients, paving the way for practical applications in radiation physics. However, a limitation arises from the constraints of conventional DNN models, which restrict the analysis to one output parameter at a time. This limitation prevented the examination of other vital parameters, such as mass absorption coefficients, compounds, and mixtures. Future research should focus on incorporating and analyzing these parameters in the models to achieve a more comprehensive analysis of the physical parameters based on NIST data.

Additionally, it is essential to address the model's limitations further to enhance result accuracy. Improving the model's regression at the discontinuity points of the NIST curves and considering the volume of data in the dataset, along with the possibility of incorporating new physical parameters, are necessary steps to improve the methodology's results. Further evaluations, such as database balancing and the exploration of additional error metrics and loss functions, should also be pursued to advance the understanding of estimating mass attenuation coefficients. This work holds promise for practical applications and underscores the need for continuous enhancement and in-depth investigations in the field.

REFERENCES

[1] OKUNADE, Akintunde A. Parameters and computer software for the evaluation of mass attenuation and mass energy-absorption coefficients for body tissues and substitutes. **Journal of Medical Physics/Association of Medical Physicists of India**, v. 32, n. 3, p. 124, 2007.

[2] AKKURT, Iskender et al. Prediction of photon attenuation coefficients of heavy concrete by fuzzy logic. **Journal of the Franklin Institute**, v. 347, n. 9, p. 1589-1597, 2010.

[3] HUBBELL, John Howard. Photon mass attenuation and energy-absorption coefficients. The International **Journal of Applied Radiation and Isotopes**, v. 33, n. 11, p. 1269-1290, 1982.

[4] MCNAIR, A. ICRU Report 33-Radiation Quantities and Units Pub: **International Commission on Radiation Units and Measurements**, Washington DC USA issued 15 April 1980, pp. 25. 1981.

[5] MEDHAT, M. E. Application of neural network for predicting photon attenuation through materials. **Radiation Effects and Defects in Solids**, 2018.

[6] BOULIC, Ronan; RENAULT, Olivier. 3d hierarchies for animation. **John Wiley and Sons**, 1991.

[7] OUELLET, Robert G.; SCHREINER, L. John. A parametrization of the mass attenuation coefficients for elements with Z= 1 to Z= 92 in the photon energy range from approximately 1 to 150 keV. **Physics in Medicine & Biology**, v. 36, n. 7, p. 987, 1991.

[8] MANJUNATHA, H. C. et al. Empirical formulae for mass attenuation and energy absorption coefficients from 1 keV to 20 MeV. **The European Physical Journal D**, v. 71, n. 9, p. 1-22, 2017.

[9] BILMEZ, Bayram et al. A comparative study on applicability and efficiency of machine learning algorithms for modeling gamma-ray shielding behaviors. **Nuclear Engineering and Technology**, v. 54, n. 1, p. 310-317, 2022.

[10] TEKIN, Huseyin Ozan et al. Validation of MCNPX with experimental results of mass attenuation coefficients for cement, gypsum and mixture. **Journal of Radiation Protection and Research**, v. 42, n. 3, p. 154-157, 2017.

[11] SINGH, V. P. et al. Determination of mass attenuation coefficient for some polymers using Monte Carlo simulation. **Vacuum**, v. 119, p. 284-288, 2015.

[12] PEDREGOSA, Fabian et al. Scikit-learn: Machine learning in Python. **the Journal of machine Learning research, v**. 12, p. 2825-2830, 2011.

[13] Xi, Y. Tune the hyperparameters of your deep learning networks in Python using Keras and Talos. Retrieved September 30, 2023, from https://towardsdatascience.com/tune-the-hyperparameters-of-your-deep-learning-networks-in-python-using-keras-and-talos-2a2a38c5ac31.

[14] SILVA, Gustavo Bernardes da. Aplicação da Transformada de Laplace para obtenção do espectro de raios X móvel. 2020. 13 f. TCC (Graduação) - **Curso de Física Médica, Ciências Exatas e Sociais Aplicadas, Universidade Federal de Ciências da Saúde de Porto Alegre**, Porto Alegre, 2020.

[15] PUJOL, João Carlos Figueira; PINTO, João Mário Andrade. A neural network approach to fatigue life prediction. **International Journal of Fatigue**, v. 33, n. 3, p. 313-322, 2011.

[16] AKKURT, Iskender et al. Prediction of photon attenuation coefficients of heavy concrete by fuzzy logic. **Journal of the Franklin Institute**, v. 347, n. 9, p. 1589-1597, 2010.

[17] EFTEKHARI ZADEH, E. et al. Application of artificial neural network in precise prediction of cement elements percentages based on the neutron activation analysis. **The European Physical Journal Plus**, v. 131, n. 5, p. 1-8, 2016.

[18] OSMAN, Gencel. The application of artificial neural networks technique to estimate mass attenuation coefficient of shielding barrier. **International journal of physical sciences**, v. 4, n. 12, p. 743-751, 2009.

[19] KUCUK, Nil et al. Modeling of gamma ray energy-absorption buildup factors for thermoluminescent dosimetric materials using multilayer perceptron neural network: A comparative study. **Radiation Physics and Chemistry**, v. 86, p. 10-22, 2013.

[20] KLEIN, Oskar; NISHINA, Yoshio. On the scattering of radiation by free electrons according to Dirac's new relativistic quantum dynamics. **Journal of Physics**, v. 52, n. 11, p. 853-868, 1929.

[21] ARCHER, Benjamin R.; WAGNER, Louis K. Determination of diagnostic x-ray spectra with characteristic radiation using attenuation analysis**. Medical physics**, v. 15, n. 4, p. 637-641, 1988.

[22] KETKAR, Nikhil; KETKAR, Nikhil. Introduction to keras. Deep learning with python: a hands-on introduction, p. 97-111, 2017.

[23] BRAHME, Anders. **Comprehensive biomedical physics**. Newnes, 2014.

[24] TURTURICA, G. V. et al. Effective Z evaluation using monoenergetic gamma rays and neural networks. **The European Physical Journal Plus**, v. 135, n. 2, p. 1-13, 2020.

[25] BRAHME, Anders. **Comprehensive biomedical physics**. Newnes, 2014.

[26] TALOS A. **Hyperparameter Optimization for Keras, TensorFlow (tf.keras) and PyTorch**. Available at: http://github.com/autonomio/talos 2020

[27] CHOLLET, Francois. **Deep learning with Python**. Simon and Schuster, 2021.