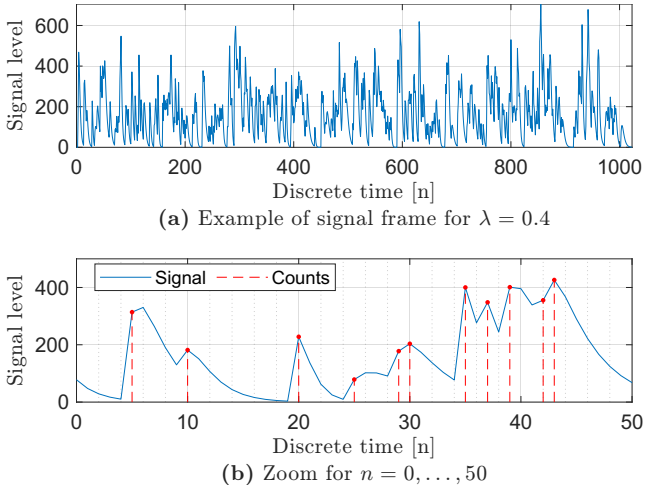
Deep Learning Based Method for Activity Estimation from Short-Duration Gamma Spectroscopy Recordings

Tom Trigano *Member* and Dima Bykhovsky *Senior Member*

***Abstract*—Gamma spectroscopy is a technique for identifying radioactive sources and evaluating their activity. Reliable estimation of activity levels remains challenging due to various physical effects, such as the pile-up effect and dead time. In numerous applications, it is critical to provide an accurate estimation of the source activity from short-duration signals. This study compares three deep neural network (DNN) architectures: Unet, Long Short Term Memory (LSTM), and an application-tailored convolutional neural network (CNN). Due to the lack of annotated datasets in the field, experiments are performed on simulated yet realistic signals and demonstrate that the application-tailored CNN outperforms both Unet and LSTM architectures. We show that the provided activity estimators have a lower variance than current state-of-the-art methods, which makes the approach well-adapted to analyze short-time signals. However, results on real signals illustrate the necessity of fully simulating pulses to match real pulse shapes, indicating that current approaches still face challenges in practical applications for a large class of detectors.**

***Index Terms*—activity estimation, deep learning, nuclear spectroscopy**



(a) Example of signal frame for $\lambda = 0.4$

(b) Zoom for $n = 0, \ldots, 50$

Fig. 1: Example of a signal for $\lambda = 0.4$: (a) raw signal, (b) zoom of the signal, with the arrival times outlined.

I. Introduction

A. General Overview

The study of gamma spectroscopy involves the analysis of radioactive sources. The field focuses on two main objectives: identifying the radionuclides present in the source through examination of the energy spectrum, which is a histogram of recorded particle energies and estimating the source's activity, also referred to as counting-rate estimation [1]. It has numerous applications, e.g., in geochemistry [2], radioactive waste management, or medical imaging [3].

Spectroscopic signals are commonly modeled as a trajectory of a compound Poisson process with some intensity λ. Within this model, a signal is made of random pulses with varying shapes. Figure 1 provides a visual example of both the detector signal and the particle presence times for the specific activity. When the activity is high, as in Fig. 1, the pulses may overlap, creating what is known as the *pile-up* effect. This effect poses a significant challenge in accurately estimating the activity as well as distorting the energy spectrum of the resulting data.

Two main approaches to contend with the pile-up effect are sensor improvements (e.g., [4], [5]) and algorithmic approaches that are used to compensate for the sensor imperfections with sophisticated machine learning (ML) solutions. An example of the ML approach is presenting the pile-up effect as a sparse regression problem [6]–[9] that was proved to be effective for activity estimation. However, the efficiency of these methods decreases as the recorded time signals are shorter or when the activity of a source increases.

The nuclear spectroscopy literature has recently suggested deep neural network (DNN) based methods, e.g. for radionuclides identification [10]–[18]. These works showed that DNNs provide promising results in identifying mixtures compared to classical ML methods. Though the application of DNNs in the field is promising, two issues prevent their use on a larger scale. The first issue is the total absence of annotated datasets in the field. Thus, all the training procedures rely on simulated data [12], [16], [19]. Besides the obvious fact that the simulated data should comply with commonly observed signal shapes and dynamics, it makes the investigation on generalization of proposed algorithms complex. The second issue is that all the network architectures described in the references above use energy spectra as inputs; however, a large number of clean electrical pulses is needed for a reliable histogram computation.

B. Main Contributions

The primary objective of this paper is to explore the effectiveness of DNN architectures for estimating counting rates. It particularly focuses on evaluating the suitability of these mod-

Tom Trigano and Dima Bykhovsky are with the Department of Electrical Engineering, Shamoon College of Engineering, Israel.

Manuscript sent July 4, 2024.

els for processing short-duration pre-recorded signals in the time domain. In nuclear spectroscopy, prioritizing short signals is often a necessity due to the high costs and extensive time requirements of laboratory measurement campaigns. Utilizing minimal data for source identification and intensity estimation not only reduces computational load but also enhances the efficiency of data processing. In particular, numerous practical applications (e.g., radiation monitoring and nuclear forensics) and theoretical ones (e.g., identification of transient radioactive phenomena) require results from small recording window times.

The main highlights of the proposed research are as follows:

1) Contrary to the current research in this area, this paper suggests using the output signal of a detector (Fig. 1a) directly, bypassing the need for histogram computation.
2) DNNs are employed to create an intermediate internal projection representation of the signal, allowing for the detection of *each* gamma particle based on signal shape, the resulting arrival times being used.
3) We illustrate that the proposed method provides a reliable activity estimate with very short-time signals, even with relatively high activity levels.
4) The paper also discusses potential limitations of applying the approach directly to real datasets, especially for higher activity levels, despite promising results for low activities.

The estimation method investigated in this paper is based on a two-step pipeline, elaborated in [20], that is visualized in Fig. 2.

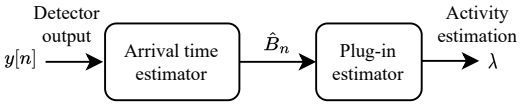


Fig. 2: Two-step activity estimation pipeline.

DNNs in Fig. 2 are used to generate a binary vector B_n whose entries equal '1' when at least a particle is present between two specific sampling points of the signal and '0' otherwise, which allows the estimation of the source's activity (using a plug-in estimate) while providing additional insight into the position of individual electrical pulses, which can be helpful for directly applying pile-up rejection or correction techniques at a later stage.

We show that a DNN-based counting rate estimation used on signals in the time domain has a lower variance and outperforms existing activity estimators relying on the same data. The bias appearing in existing plug-in estimators for high activities is learned and corrected, which makes the approach suitable for short-duration signals (of the order of hundreds of microseconds). The performance evaluation shows a significant improvement in the DNN-based optimized pipeline results compared to the previously published results in [20], [21].

II. Signal Model and Dataset Generation

A. Model and Signal Generation

Mathematically, the continuous-time output signal from the detector, $y(t)$, can be expressed as follows:

$$y(t) = \sum_{n=0}^{\infty} E_n h_n(t - T_n) + \varepsilon(t), \tag{1}$$

where the sequence $\{T_n,\ n \geq 0\}$ is a sample path of a homogeneous Poisson process with unknown normalized intensity λ, which represents the arrival times of the particles, $h_n(\cdot)$ is the shape of the electrical pulse associated with the n-th impinging particle, E_n is its energy and $\varepsilon(t)$ is additive noise. In the simulated signals used for the learning procedure, the shapes $h_n(t)$ are modeled by truncated Gamma shapes with parameters α and β, as described in detail in previous contributions of the authors [6], [20], as follows:

$$h(t) = t^{\alpha} e^{-\beta t} \quad 0 < t < T. \tag{2}$$

This model makes it possible to simulate electrical pulses with a rapid rise, followed by a slow exponential decay, as typically encountered in high-purity Germanium (HPGe) detectors. This signal is sampled at Δt, which denotes the sampling period to produce the discrete-time signal $y[n]$. The method described in the paper aims to provide a reliable intensity estimate λ given $y[n]$.

Simulated signals were generated from a homogeneous Poisson process with varying true intensity λ from 0.05 to 0.6, representing a broad range of radioactive activity. The energies E_n followed a mixture of two Gaussian densities with means equal to 100 and 225 and variances of 10 and 15. The noise variance σ was set to 1, reflecting a typical signal-to-noise ratio encountered in practice. In the given experiments, a random shape was chosen from 20 Gamma shapes to illustrate the variability of the generated electrical pulses for each point of the Poisson process. The resulting signals are based on Eq. (1). Further details on the generation of the signals can be found, e.g., in [20].

B. Performance Validation

The goal of the performance validation is to identify the generalization properties of the proposed solutions. We used seven normalized activities $\lambda = 0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.65$ for training, validation and testing. There were an additional six activities of $\lambda = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6$ that were used only for validation and testing (but not for training) in order to validate the generalization properties of the proposed solution. Train and test λ values of the dataset are visualized in Fig. 3.

The training dataset was comprised of 10000 signals of 1024 samples for each activity level. The validation and test datasets consist of 500 signals each, with the same length for every activity level. Note that 1024 sampling points correspond to very short-duration signals; for example, on the real data used in this paper, sampled at $10\,\mathrm{MHz}$, it means that we estimate the activity over a period of $102.4\,\mu\mathrm{s}$.

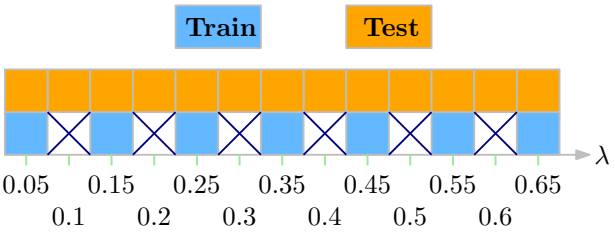


Fig. 3: λ values for train and test datasets. In order to check the generalization for different λ, some values are present only in the test dataset but not in training.

III. Description of the Arrival Time Estimators

The arrival time estimator aims to determine whether one or more particles can be attributed to each time sample of $y[n]$. The resulting output is a binary decision for each n. Three DNN architectures were evaluated in this paper for arrival times estimation, each related to a different problem formulation approach.

A. U-Net Network Architecture

The presence or absence of particles between each sampling point of $y[n]$ may be considered a two-class (binary) signal segmentation problem. The U-Net architecture [22] is one of the most common segmentation solutions; it is a classical convolutional neural network (CNN), initially proposed for biomedical image segmentation, with inherent encoding and decoding structures. Down-sampling comprises multiple continuous convolution layers, and the corresponding up-sampling comprises multiple continuous convolution and deconvolution layers. Residual concatenation-based skipping is used between the corresponding layers. We used a one-dimensional (1D) version of U-Net, schematically presented in Fig. 4. The number of filters in each following downsampling block is twice that of the previous block.

The implemented network (Fig. 4) was configured with 2 to 4 down-sampling and up-sampling block pairs with filter numbers of 8, 16, 32, and 64, correspondingly. The kernel length of all the filters was set to 3, with causal padding.

B. Long Short-Term Memory (LSTM) Network Architectures

Another approach considers the signal's sequential aspect. In this case, the problem may be classified as a *sequence-to-sequence* problem with input and output sequences of the same length (Fig. 5a). The resulting architecture is based on bidirectional LSTM layers. To learn feature representation at LSTM inputs, one or more repeating inception *Conv-1D* modules [23] were used. Features were fed in the bidirectional LSTM layer concurrently with the dense layer, resulting in a binary prediction of activity presence for each sample in a frame. The network architecture is outlined in Fig. 5b.

The implemented network (Fig. 5b) was configured with one- or bi-directional LSTM layers with either 8, 16 or 32 units each. The number of successive inception modules was between 1 and 3, with 4, 8 or 16 filters of each size in an inception block.

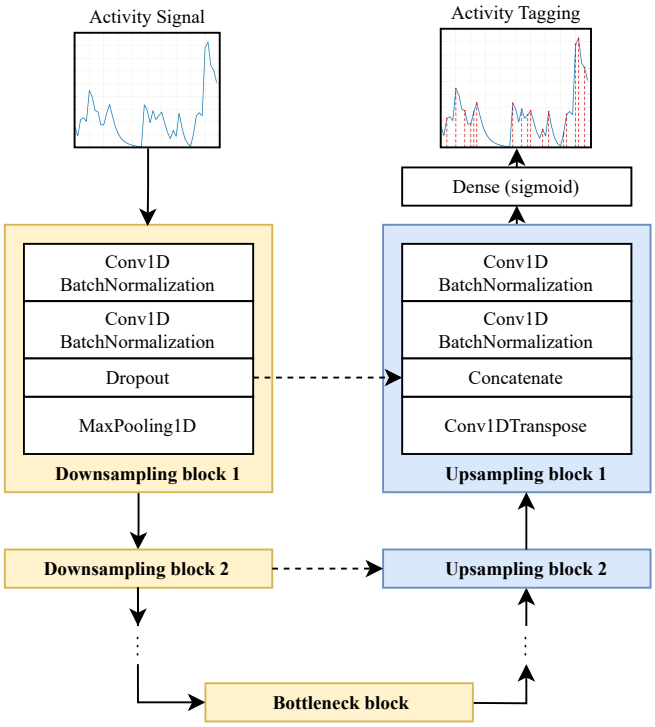
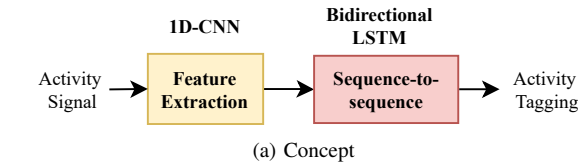
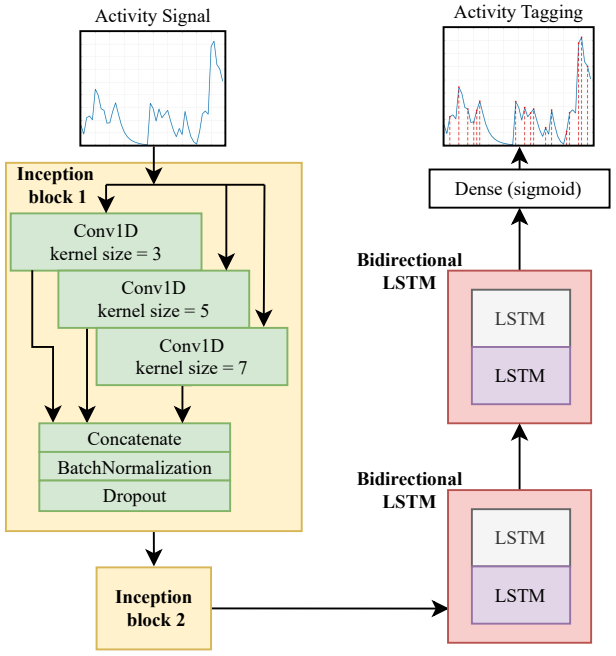


Fig. 4: U-Net architecture.

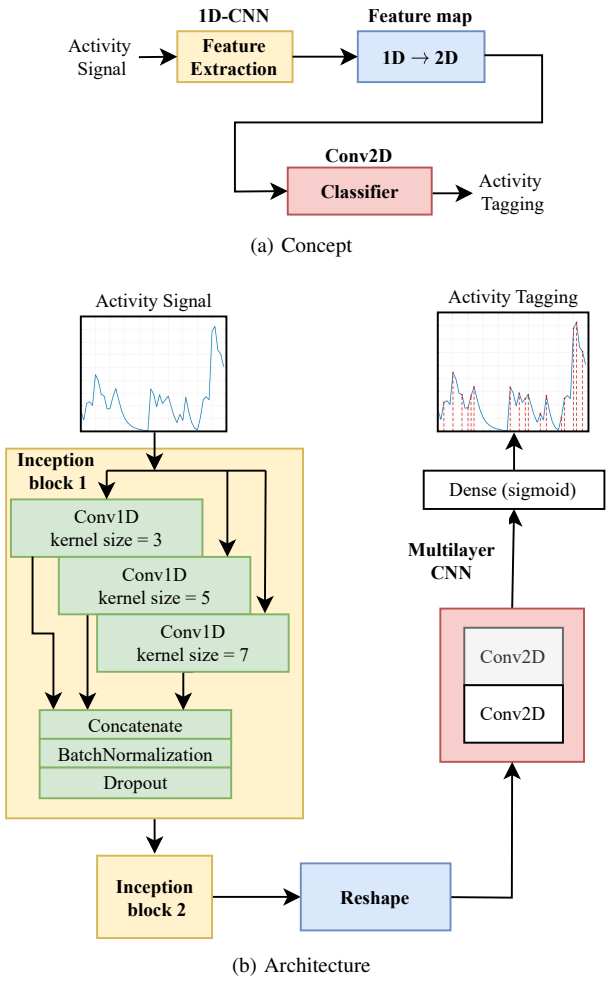


(a) Concept



(b) Architecture

Fig. 5: LSTM architecture: (a) concept and (b) architecture.



(a) Concept

(b) Architecture

Fig. 6: CNN architecture: (a) concept and (b) architecture.

C. Classification CNN Architecture

This method generates features by one or more inception Conv-1D modules, as in the LSTM architecture above. However, the resulting features are reshaped into two-dimensional (2D) feature embeddings. These embeddings are then fed into a 2D CNN-based classifier (Fig. 6a). A schematic presentation of the network architecture is presented in Fig. 6b.

The implemented network (Fig. 6b) was configured with 1 to 2 2D-CNN layers. The number of successive inception modules was between 1 and 3, with 4, 8 or 16 filters of each size in an inception block.

D. Loss Functions

In this specific framework, the choice of the loss function used for optimizing the networks' weights may have a significant impact on the estimation performance. Since the arrival times appearing in Eq. (1) are distributed based on a Poisson process, the data used is naturally imbalanced, and the degree of imbalance highly varies depending on the activity. The number of labels equal to '0' will dominate for low activities. Conversely, the labels will be mostly equal to '1' for very high activities. Following these considerations, we evaluated three kinds of loss functions:

1) The binary cross-entropy (BCE) loss that is commonly used for classification problems.
2) The focal-loss [24], which addresses class imbalance during training by focusing learning on hard misclassified examples.
3) The Poisson loss [25] that is used for regression problems where the target variable is a count variable, such as the number of events or occurrences in a given time or space.

E. Learning Parameters

1) Learning Rate (LR): We started with an LR of 0.001, which is a standard starting point for many DL tasks. This rate was adjusted dynamically using an LR scheduler based on the plateau in validation loss, reducing it by a factor of 0.5 when no improvement was seen for 5 consecutive epochs.

2) Batch Size: A batch size of 32 samples was used. This size was a balance between computational efficiency and model stability during training, making it suitable for our hardware constraints and the size of the training dataset.

3) Epochs: The models were trained for up to 100 epochs, but we employed early stopping based on the validation loss to prevent overfitting and unnecessary training. Training was stopped if the validation loss did not improve for more than 5 epochs.

IV. Activity Estimator Based on Estimated Arrival Times

A. Theoretical Estimator

The first activity estimator approach relies on an existing estimator presented in [21], which provides an unbiased activity estimate. We denote this method by "theoretical estimator" in the following. This estimator is defined as

$$\hat{\lambda} = -\frac{1}{\Delta t} \ln\Big(1 - \frac{1}{N} \sum_{n=0}^{N-1} \hat{B}_n\Big), \tag{3}$$

where N denotes the number of sampling points of a signal $y[n]$, and $\hat{B}_n$ is a binary indicator variable, equal to '0' if no T_n values are present between the n-th and the $(n+1)$-th sampling points, and to '1' otherwise. As shown in [21], this estimator is asymptotically unbiased and statistically consistent. However, as its convergence to the true activity is asymptotic, its quality depends on the total number of available sampling points. Therefore, when dealing with very short-duration signals, the estimator derived from (3) is highly sensitive to the $\hat{B}_n$ estimation accuracy. Therefore, we suggest learning the functional relation between B_n and λ to circumvent this issue, as detailed below.

B. Learned Estimator

A straightforward plug-in of the DNN's output in the theoretical estimate from the previous section yields a systematic underestimation of the activity. In order to improve the source activity estimation results based on the DNN's outputs, we have proposed an improved activity estimator based on a

learned functional of the average number of active sampling points, as presented below.

The theoretical estimator is an analytical function of the average number of particle presence indicators,

$$\mu = \frac{1}{N}\sum_{n=0}^{N-1}\hat{B}_n. \tag{4}$$

In order to improve the overall pipeline performance (Fig. 2), we aim to correct the bias introduced by the fact that there may be more than one particle arrival between two sampling points of the signal. Consequently, we model the plug-in estimator by an m-th order polynomial of the form

$$p(\mu) = \sum_{k=0}^{m} a_k \mu^k, \tag{5}$$

where the coefficients $a_k > 0$ are learned as a minimum mean square estimation (MMSE) fit between "oracle" activity λ and the value of $p(\mu)$. From the physical nature of the signal $y[n]$, the resulting $p(\cdot)$ function values on the corresponding support values of λ are non-decreasing, resulting in one-to-one mapping between μ and λ.

V. Results on Synthetic Data

A. Evaluation Protocol

We conducted simulations with the goal of evaluating the performance of the activity estimation pipeline (Fig. 2) for different combinations of methods for arrival time and activity estimation. We compared the following estimation approaches:

1) Oracle estimator: The oracle estimator is the standard maximum likelihood estimate defined as follows:

$$\hat{\lambda}_{oracle} = \frac{M}{T_{max}}, \tag{6}$$

where M is the exact number of electrical pulses in the recorded signal and T_{max} is the arrival time of the last electrical pulse.

2) LASSO based estimator: The so-called LASSO based estimator is introduced in [21]. The arrival times are estimated with a post-processed version of the LASSO regressor introduced in [26], and are plugged into Eq. (3) for the estimation of the activity. This method provides good estimates of a normalized activity of up to $\lambda = 0.5$ and is used for the sake of comparison with a state-of-the-art method.

3) DNN with theoretical plug-in estimator: The post-processed version of LASSO from [21] is replaced with a DNN that estimates the active bins B_n (Sec. III), coupled with the estimator in Eq. (3).

4) DNN with learned plug-in estimator: The proposed algorithm learns the plug-in estimator directly from the DNN-produced B_n values, following Eq. (5).

5) Optimized pipeline: Arrival time estimation is based on the binary decision concept. However, the output of a typical DNN is a probabilistic estimate. This estimate is converted to a binary decision by comparing it to a pre-defined threshold, typically 0.5. Nevertheless, this threshold may be further optimized. The threshold is optimized numerically to improve algorithm performance.

The evaluated simulation combinations presented in the paper are summarized in Table I.

The selection of comparative methods is motivated by the following considerations: the oracle estimator, representing the best theoretical estimator, relies on knowing the exact number of electrical pulses and their last arrival time. Although this information is not practically available, the oracle estimator sets the benchmark for the lowest possible estimation variance, serving as a gold standard. The LASSO-based estimator, a recent development for estimating source activity over short segments, is included to provide a contemporary comparison point. Finally, the proposed method includes several enhancements, such as the use of deep neural networks (DNNs), a learned plug-in estimator, and an adaptive threshold, as detailed above. It is essential to compare the performance of these estimators to clearly demonstrate the contributions of each component to the overall method.

B. DNN Performance

Experiments were performed on a Intel Core i7-10700 computer, with a 2.90-GHz processor CPU, 64 GB of RAM, and an Nvidia RTX A4000 16GB GPU. We used Python 3.9 with Tensorflow 2.10. We calculated the accuracy, sensitivity (recall) and specificity for the investigated architectures on the test set for all the configurations.

The scatter plot shown in Fig. 7 displays the sensitivity and specificity performance of all the architectures discussed in this paper. The plot consists of multiple points, each representing a unique combination of hyperparameters and activities. While there are no significant variations between the architectures and losses, we can see that all the architectures have very similar performance for low activity. However, the LSTM and CNN architectures perform better for higher activity. Therefore, it is recommended to choose either the CNN or the LSTM architecture for activity estimation. Further details about the performance of the various architectures are provided in the Appendix.

Due to the similarities in the activity estimation results across different architectures, the following results are presented for a single example (LSTM architecture implementation, 1-layer inception, 16 filters, bi-directional LSTM layer with 32 units) for brevity.

A comparison between method #1 and method #3 is presented in Fig. 8. The results show that the DNN solution yields excellent results for low activity and has an inherent bias that increases as activity grows. We emphasize that as the activity of the source increases, so does the probability of having one or more photon arrivals between two sampling points of the signal, making the accurate estimation more difficult to attain. This shows, *a posteriori*, that the DNN approach has no trouble finding whether one arrival or more occurred between two sampling points, but struggles to estimate the exact number of pulses in a segment as the activity grows.

We next compare the performance of method #2 and method #3 in Fig. 9. The results are displayed in the activity range $[0.1, 0.4]$, since results from [20] and [21] were available in this range only. The results obtained with the proposed

TABLE I: List of the evaluation performance combinations

#	Arrival Estimator	Plug-in	Note
1	Oracle	Maximum Likelihood	Best estimator, unattainable in practice
2	Post-processed LASSO	Eq. (3)	Concurrent method
3	DNN	Eq. (3)	
4	DNN	Learned (Eq. (5))	
5	DNN	Learned (Eq. (5))	Pipeline optimization

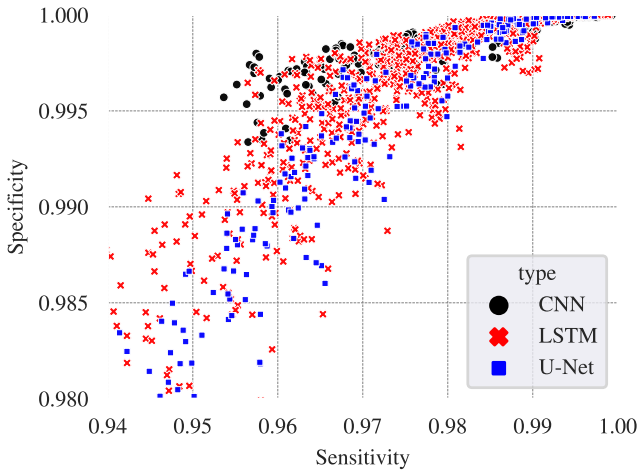


Fig. 7: Sensitivity and specificity for all architecture hyper-parameters and activities. LSTM and CNN architectures have better performance for higher activity levels.

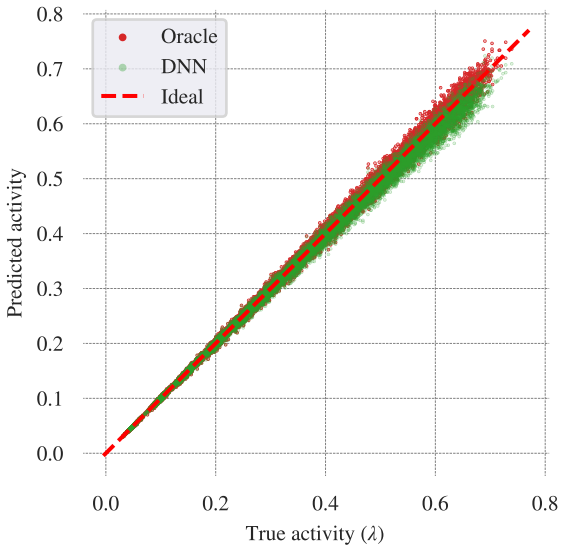


Fig. 8: Illustration of the performance of DNN-based estimator paired with an optimal plug-in estimator. The bias of the DNN estimator is clearly notable.

DNN-based approach show a bias when compared to the method proposed in [20], and, more specifically, the average value of the activity estimate is always slightly lower than its counterpart. However, in contrast to the alternative method, our approach has a much lower variance, and the exact value of the activity always falls within the 90% confidence interval. Therefore, the DNN-based method provides a more reliable estimate than its unsupervised counterpart.

C. Learned Plug-in Performance

In order to learn the coefficients of the plug-in estimator in Eq. (5), the following steps were taken:

1) The DNN estimator was trained using the training data from the database.
2) The validation database was used to learn the relation ('learned' trendline) between the resulting B_n values and the actual ones. The polynomial order was set as $m = 3$; higher values provided no improvement.
3) The 'test' database was used for performance evaluation.

The plug-in learns μ to $\lambda = p(\mu)$ mapping to provide an improved estimate for each μ value. Figure 10 illustrates the learned plug-in results. This plug-in's main advantage lies in its removal of the inherent bias resulting from the DNN estimator.

D. Threshold Optimization

During our evaluation, we noticed that the probabilistic DNN's output has an asymmetric distribution and varies among different activities. As an example, the histograms of the prediction probabilities for B_n values for $\lambda = 0.1$ and 0.6 are presented in Fig. 11.

Clearly, the optimal threshold for the estimation of B_n strongly depends on λ itself. Nevertheless, optimizing the overall pipeline by averaging performance across all activities in the validation database is possible. The root mean square error (RMSE) of the λ estimation for different threshold values is shown in Fig. 12, and this threshold is used in method $\#5$.

The corresponding receiver operating characteristic (ROC) curve for all the detected probabilities of B_n values, along with the corresponding default and optimal threshold values, is presented in Fig. 13. It shows that the optimal threshold value represents a compromise between the false positive rate (FPR) and the true positive rate (TPR).

The examples of a confusion matrix for method $\#5$ are presented in Fig. 14. The performance of the proposed method is illustrated in Fig. 15; the resulting estimator is unbiased and performs very close to the performance of the oracle estimator.

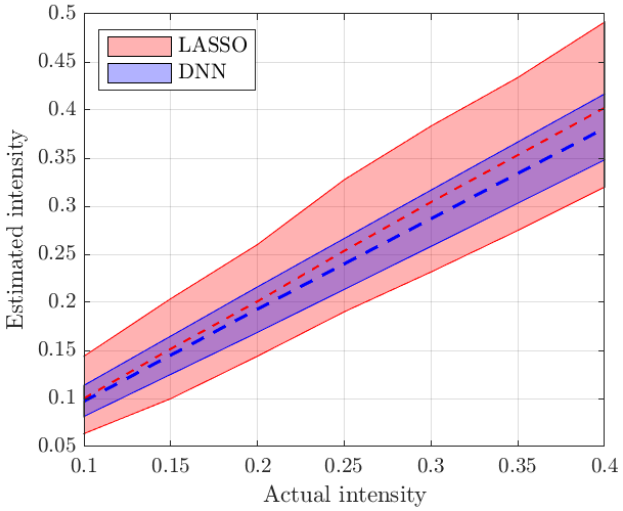


Fig. 9: Comparison of methods #2 (red) and #3 (blue) for arrival time estimation (dotted line: obtained average values, shaded areas: 90% confidence intervals).

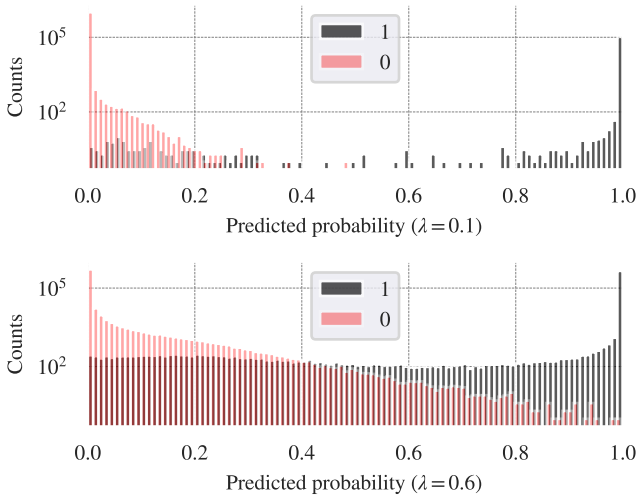


Fig. 11: Histogram of the predicted probabilities of B_n for two different activities.

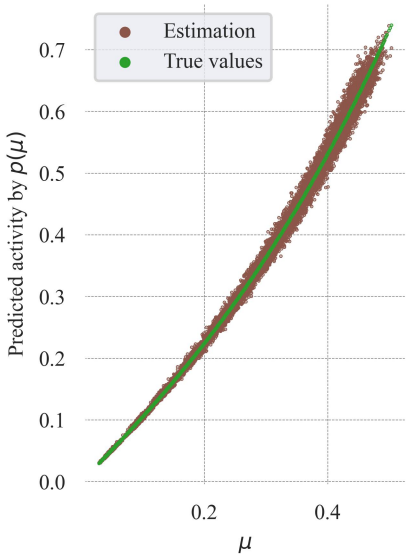


Fig. 10: Learned plug-in estimate with each μ value mapped to the corresponding activity by Eq. (5).

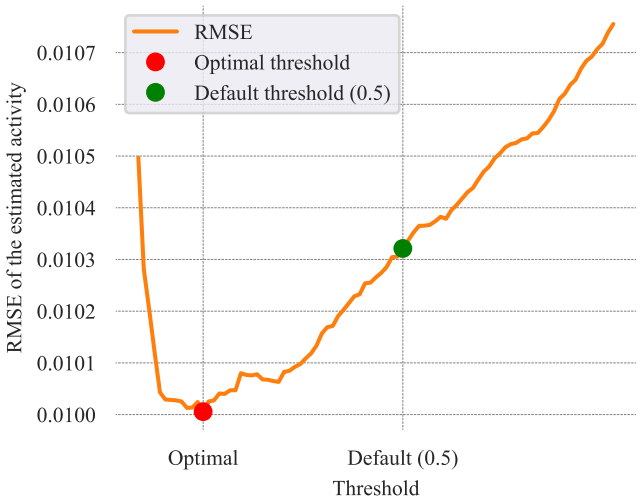


Fig. 12: Influence of threshold values on the RMSE performance of the activity estimation averaged over all activities.

Finally, we present the averaged RMSEs of the four methods (#1, #3, #4, and #5) for different activity levels in Fig. 16. We can observe that the results obtained by method #5 are excellent and close to optimality.

VI. Results on Real Datasets

A. General Signals

We investigated the performance of the networks trained on the signals generated based on Eq. (1) on four real signals, based on method #3. All the measurements were performed at CEA Saclay on radioactive mixtures with normalized activities of 0.04, 0.05, 0.1 and 0.3, respectively. In order to investigate the proposed methods on short-duration signals, we use 200 segments of only 1024 sampling points. Each segment provided an estimate of the activity, based on equation (3).

Results of the estimated activities are displayed as 90% confidence intervals in Fig. 17. We can observe that there is a decade between the true activity and the estimated one, which might seem disappointing. However, an encouraging fact is that (at least for low activity) the scale seems constant for all the considered activities, as seen in Fig. 18. This last point indicates that the trained DNN estimates single peaks with multiple active peaks. Indeed, the peaks simulated to train the simulators are not well suited to the real recorded signals, in terms of both shape and associated energies. Consequently, large electrical pulses or pulses with high amplitude can easily mislead the trained DNN, which will return several active times in place of a single one, as seen in the time domain in Fig. 19. This difficulty was also noted in previous

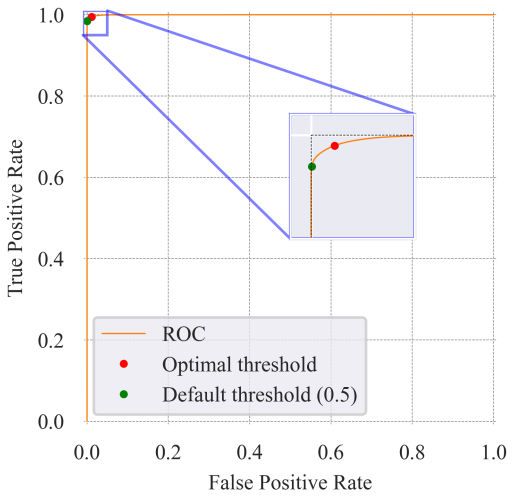


Fig. 13: The location of the default and optimal threshold on the ROC curve.

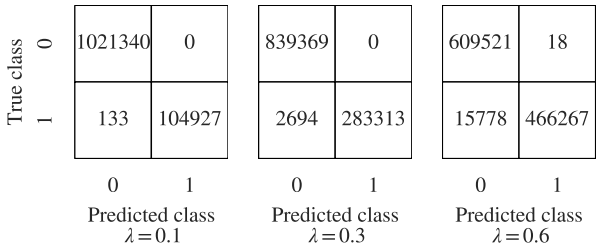


Fig. 14: Confusion matrix of a selected λ values for DNN with the optimal threshold values.

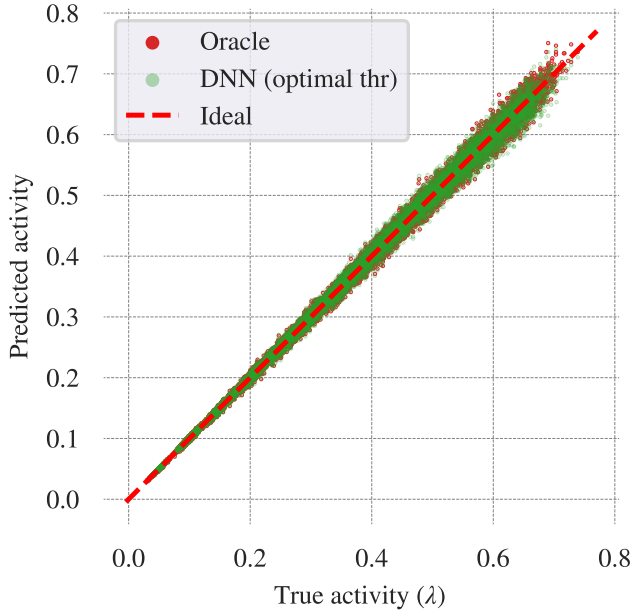


Fig. 15: Visualization of DNN performance with learned plug-in and threshold optimization compared to "oracle" performance. Both plots are very similar.

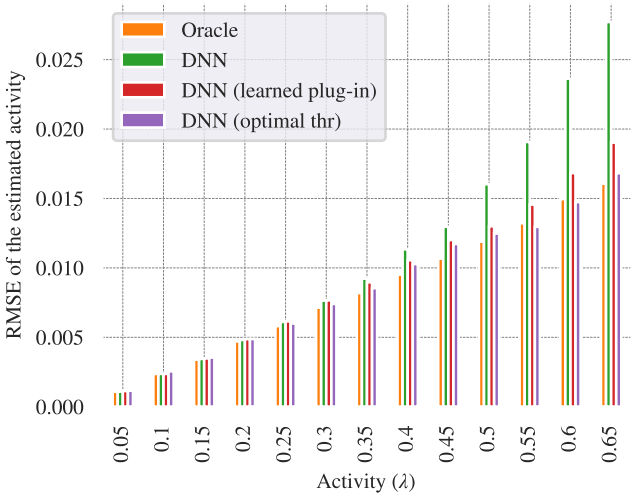


Fig. 16: RMSE performance comparison for network-based methods. High RMSE values for DNN are due to high estimation bias that is removed by the learned plug-in estimator.

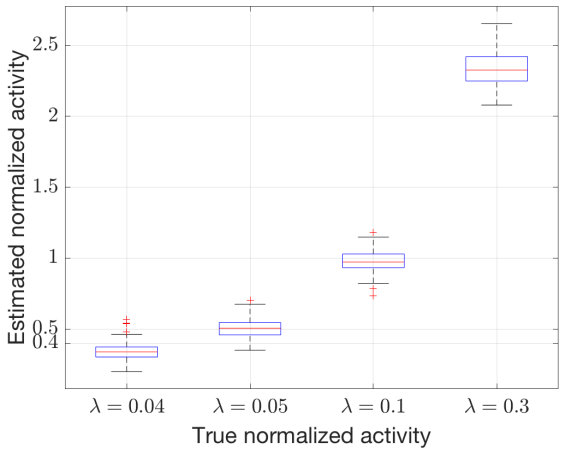


Fig. 17: Activity estimation results on real recordings. 90% of the estimated activity (y-axis) fall within the whisker plot.

contributions [6], [20], and is the most problematic when using a time-domain-based approach for activity estimation. The proposed approach is based on the supervised learning paradigm, and simulator inaccuracies may somewhat alter the resulting performance.

When normalizing the average of B_n in (3) by the average number of active times to model single electrical pulses, we can notice that the obtained estimations are more consistent with the true values, as illustrated in Fig. 20. Although the normalization introduced is not a practical solution and cannot replace a better-trained network, it shows that the overall approach provides encouraging results on real data, even if the associated DNN was poorly trained.

B. Results on Data-Driven Simulated Signals

As seen in the previous section, the localization of individual peaks is made harder when the simulated electrical

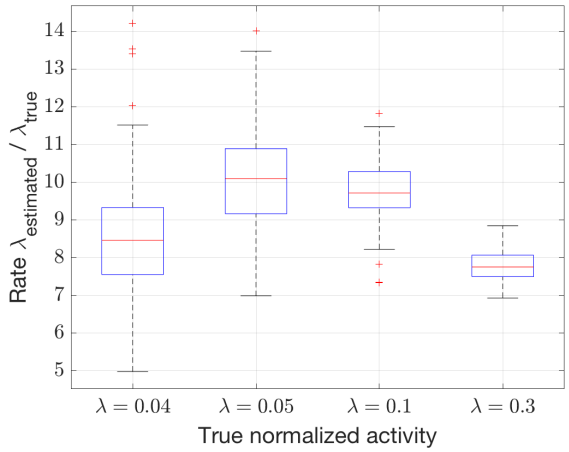


Fig. 18: Distribution of the ratio between estimated and true activities. We note that for all activities, the ratio is approximately tenfold.

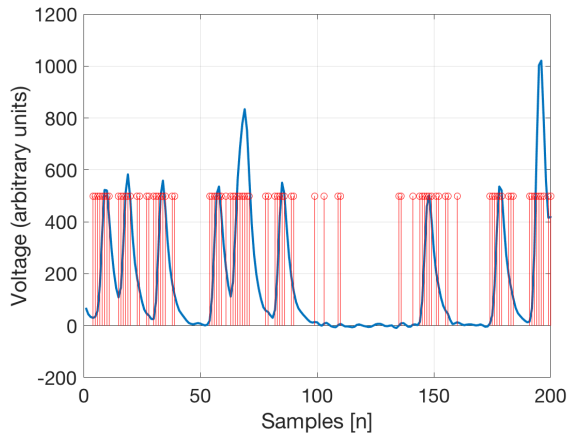


Fig. 19: Example of real-time signal (blue) and associated predicted arrival times B_n (red).

pulses' shapes are not well suited to the real data. Indeed, the shapes of electrical pulses from real recordings strongly depend, among other things, on the impact point of the particle, the temperature and the detector's geometry, most of these parameters being either unknown or unavailable in the recordings. However, the pulses' shapes of the simulator could be learned from a real dataset to make the approach more robust. To better illustrate the connection between the simulated signals and the DNNs' performance, the described networks were trained on simulated signals, whose Gamma shapes were computed based on a real signal, and activities were estimated using method #3. We used a raw HPGe measurement signal of a mixture of Cesium and Americium sources as a reference for gamma-shape-based simulation.

After discarding electrical artifacts common in the field (e.g. saturated peaks, detectors resets and baseline), single electrical pulses were extracted from one second of the signal, based on the analysis of their shape, and after discarding shape with abnormally low or high energies, including pile-ups.

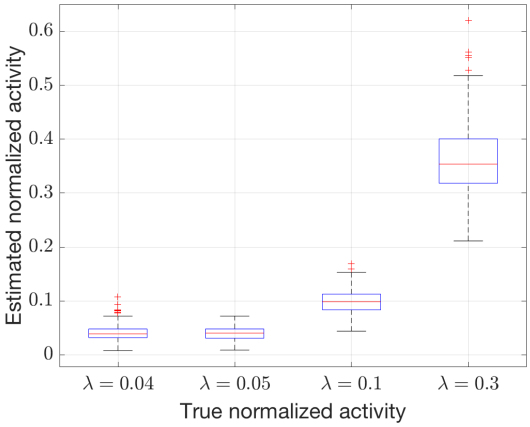


Fig. 20: Activity estimation results on real recordings after normalization. There is a better adequacy between the true and estimated activities, though the variance at high activities remains high.

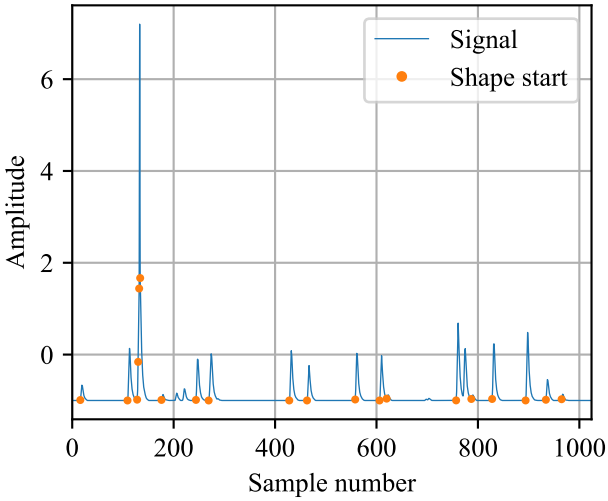


Fig. 21: Example of real electrical signal (blue) and estimated particle arrival times (orange dots).

The resulting signal included about 9×10^4 shapes.All these single pulses were estimated by Gamma shapes with different parameters by minimum mean square error (MMSE) fit[1], and these Gamma shapes were used to simulate electrical signals to train the described DNNs. An example of an real signal with identified shapes is presented in Fig. 21. The results show a resemblance between the visual inspection and evaluation results, illustrated that activity estimation can be achieved provided the simulated signals used for training have a relative adequacy with the real recorded electrical signal.

C. Discussion

Our results with synthetic signals demonstrate the potential of our DNN-based approach for estimating activities in short-

[1]Additional details on signal adjustments are available online at Github repository github.com/bykhov/activity_estimation.

duration signals. However, testing with real data highlights areas for improvement. The discrepancy between simulated and real-world signals results in a performance gap. For instance, when we adapt the pulse shapes to those derived from real data, as shown in Fig 21, we achieve more accurate counting results. This adjustment suggests that real data integration could significantly improve model accuracy.

Similarly, our simulations do not account for certain signal artifacts like resets and detector saturation, and they do not offer enough examples for robust training. We plan to address these issues in future work by refining the alignment between experimental measurements and simulations used for DNN training.

VII. Summary and Future Work

This paper presented deep learning architectures for solving the problem of activity estimation in gamma spectroscopy. Results from simulations are consistent with other state-of-the-art methods relying on the time domain, but with a significantly lower variance, and the CNN architecture yields superior results in most scenarios. Simulation results are very good and promising, enabling the estimation of activity over very short-duration time segments, which is important for real-time applications. We show on real data that the proposed solution provides encouraging, even if out of scale, results, and that the labeled signal for supervised learning requires precise information on arrival time, energy and shape of each gamma particle. Since practically extracting such information is unfeasible, a Monte-Carlo simulation of such a signal is used.

Future research will include the refinement of our simulation models to more closely mirror the properties of real spectroscopic signals for a specific detector. In particular, the integration of real and simulated data in training our deep learning models should be investigated. Furthermore, the generalization properties of the proposed approach will be validated on different datasets.

Appendix A
DNN Performance

This appendix presents additional discussion regarding the performance outcomes of various DNN architectures.

1) Performance Comparison Between Different Architectures: CNN and LSTM architectures have very similar performance. Figure 22 shows an example for λ equal to 0.5. While U-Net displays lower performance, there are more advanced versions within the U-Net family of architectures that may offer better results. However, despite their increased complexity, we do not anticipate a significant improvement over CNN and/or LSTM architectures.

2) Performance for Different Activity Levels: The relationship between specificity and λ shows that as λ increases, there is a decline in performance (Fig.23).

3) Comparison of Loss Functions: In terms of performance, BCE loss is very similar to Poisson loss, while focal loss is considered to be the least effective (Fig.24).

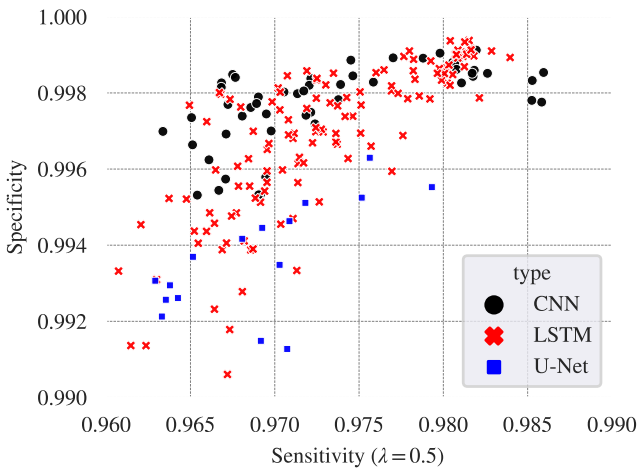


Fig. 22: Comparison of the performance of various architectures for $\lambda = 0.5$. LSTM and CNN architectures clearly outperform U-Net architecture.

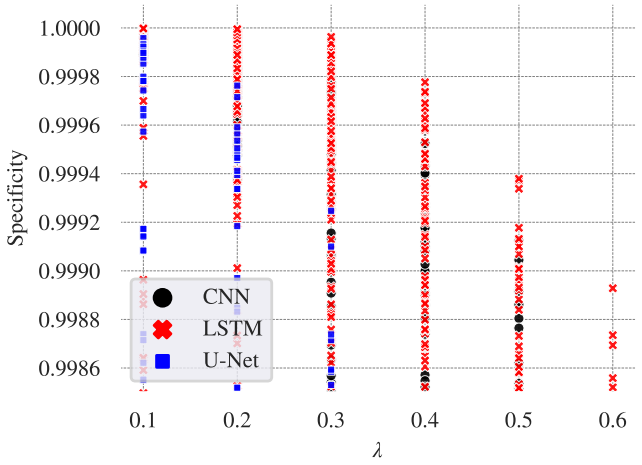


Fig. 23: Plot of specificity vs λ: as the value of λ increases, the specificity decreases, resulting in a decline in performance.

4) Complexity: Accurately evaluating the complexity of a DNN model's inference is a complex task [27]. We used the number of model parameters as a measure of complexity. Deeper implementation, which means more parameters, resulted in better performance for all models (Fig. 25).

References

[1] G. F. Knoll, *Radiation Detection and Measurement*. Hoboken, N.J: Wiley, 4 edition ed., Aug. 2010.

[2] R. Schafer and P. Schmidt, *Methods in Physical Chemistry*. Wiley, 2012.

[3] C. Baker, D. Sarno, R. J. Eckersley, and B. Zeqiri, "Pulse Pileup Correction of Signals From a Pyroelectric Sensor for Phase-Insensitive Ultrasound Computed Tomography," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, pp. 3920–3931, Oct. 2019.

[4] M. D. Haselman, J. Pasko, S. Hauck, T. K. Lewellen, and R. S. Miyaoka, "FPGA-Based Pulse Pile-Up Correction With Energy and Timing Recovery," *IEEE Transactions on Nuclear Science*, vol. 59, pp. 1823–1830, Oct. 2012.

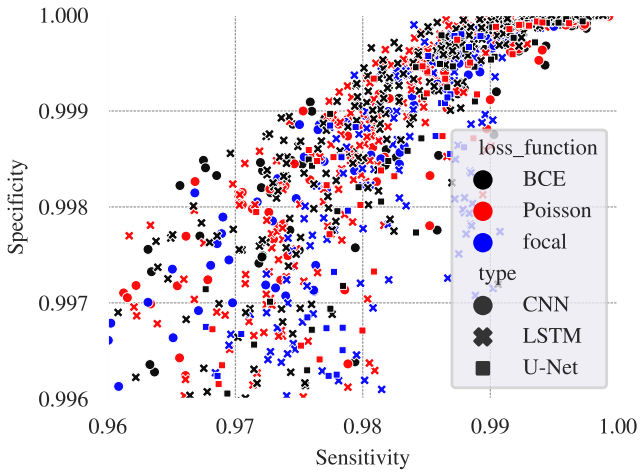


Fig. 24: Plot of specificity vs sensitivity: BCE and Poisson losses are very similar. The focal loss appears to be less effective.

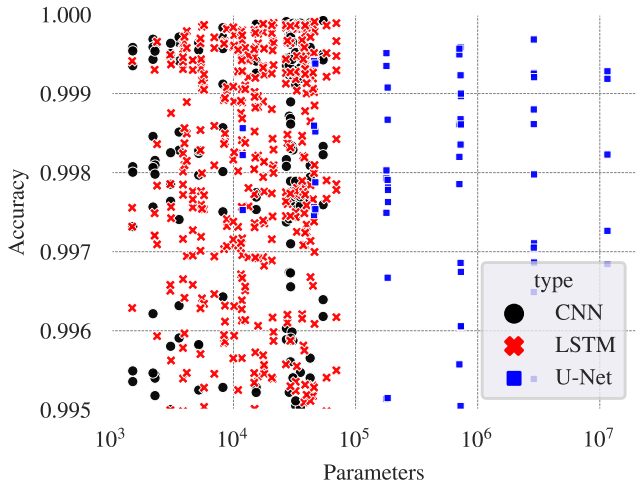


Fig. 25: Accuracy of different architectures as a function of the number of parameters. LSTM and CNN architectures have significantly lower numbers of parameters.

[5] T. Chi, M. Chen, and D. Zhang, "Implementation and performance analysis of pulse pile-up correction algorithm on DSP architecture," in *2015 IEEE 5th International Conference on Electronics Information and Emergency Communication*, pp. 372–375, May 2015.

[6] T. Trigano, Y. Sepulcre, and Y. Ritov, "Sparse reconstruction algorithm for nonhomogeneous counting rate estimation," *IEEE Transactions on Signal Processing*, vol. 65, pp. 372–385, Jan. 2017.

[7] T. Trigano and Y. Sepulcre, "Data-driven parameter selection for activity estimation in nuclear spectroscopy," *Signal Processing*, vol. 151, pp. 99–106, Oct. 2018.

[8] T. Trigano, I. Gildin, and Y. Sepulcre, "Pileup Correction Algorithm using an Iterated Sparse Reconstruction Method," *IEEE Signal Processing Letters*, vol. 22, pp. 1392–1395, Sept. 2015.

[9] B. Liu, M. Liu, M. He, Y. Ma, and X. Tuo, "Model-Based Pileup Events Correction via Kalman-Filter Tunnels," *IEEE Transactions on Nuclear Science*, vol. 66, pp. 528–535, Jan. 2019.

[10] S. M. Galib, P. K. Bhowmik, A. V. Avachat, and H. K. Lee, "A comparative study of machine learning methods for automated identification of radioisotopes using NaI gamma-ray spectra," *Nuclear Engineering and Technology*, vol. 53, pp. 4072–4079, Dec. 2021.

[11] Z. Iqbal, D. Nguyen, M. Thomas, and S. Jiang, "Deep learning can accelerate and quantify simulated localized correlated spectroscopy," *Scientific Reports*, vol. 11, p. 8727, Apr. 2021.

[12] B. Jeon, J. Kim, E. Lee, M. Moon, and G. Cho, "Pseudo-Gamma Spectroscopy Based on Plastic Scintillation Detectors Using Multitask Learning," *Sensors*, vol. 21, p. 684, Jan. 2021.

[13] M. Gomez-Fernandez, W.-K. Wong, A. Tokuhiro, K. Welter, A. Alhawsawi, H. Yang, and K. Higley, "Isotope identification using deep learning: An explanation," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 988, p. 164925, Feb. 2021.

[14] B. Ayhan and C. Kwan, "New Results on Radioactive Mixture Identification and Relative Count Contribution Estimation," *Sensors*, vol. 21, p. 4155, June 2021.

[15] A. G. L. Otero, A. J. Potiens Junior, and J. T. Marumo, "Comparing deep learning architectures on gamma-spectroscopy analysis for nuclear waste characterization," *Meeting on Nuclear Applications (ENAN), 14th, October 21-25, 2019, Santos, SP*, Aug. 2021.

[16] J. Hwang, B. Jeon, J. Kim, H. Kim, and G. Cho, "Deep learning-based spectrum-dose prediction for a plastic scintillation detector," *Radiation Physics and Chemistry*, vol. 201, p. 110444, Nov. 2022.

[17] F. Mendes, M. Barros, A. Vale, and B. Goncalves, "Radioactive hotspot localisation and identification using deep learning," *Journal of Radiological Protection*, vol. 42, p. 011516, Jan. 2022. Publisher: IOP Publishing.

[18] J. Kim, J. Hwang, G. Song, K. Ko, H. Kim, and G. Cho, "Untrained neural network-based unfolding method for quantitative analysis of NaI(Tl) gamma spectrometers," *Radiation Physics and Chemistry*, vol. 209, p. 110993, Aug. 2023.

[19] A. Turner, C. Wheldon, M. R. Gilbert, L. W. Packer, J. Burns, T. K. Wheldon, and M. Freer, "Generalised gamma spectrometry simulator for problems in nuclide identification," *Journal of Physics: Conference Series*, vol. 1643, p. 012211, Dec. 2020.

[20] Y. Sepulcre, T. Trigano, and Y. Ritov, "Sparse Regression Algorithm for Activity Estimation in γ Spectrometry," *IEEE Transactions on Signal Processing*, vol. 61, pp. 4347–4359, Sept. 2013.

[21] T. Trigano and J. Cohen, "Intensity Estimation of Spectroscopic Signals With an Improved Sparse Reconstruction Algorithm," *IEEE Signal Processing Letters*, vol. 24, pp. 530–534, May 2017.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[25] N. Fallah, H. Gu, K. Mohammad, S. A. Seyyedsalehi, K. Nourijelyani, and M. R. Eshraghian, "Nonlinear poisson regression using neural networks: A simulation study," *Neural Computing and Applications*, vol. 18, pp. 939–943, 2009.

[26] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer, 2nd ed. 2009. corr. 3rd printing ed., Feb. 2009.

[27] X. Hu, L. Chu, J. Pei, W. Liu, and J. Bian, "Model complexity of deep learning: A survey," *Knowledge and Information Systems*, vol. 63, pp. 2585–2619, 2021.