




## Article

# Recognition of Cosmic Ray Images Obtained from CMOS Sensors Used in Mobile Phones by Approximation of Uncertain Class Assignment with Deep Convolutional Neural Network

Tomasz Hachaj <sup>1,\*</sup> , Łukasz Bibrzycki <sup>2</sup>  and Marcin Piekarczyk <sup>1</sup> 

<sup>1</sup> Department of Signal Processing and Pattern Recognition, Institute of Computer Science, Pedagogical University of Krakow, 2 Podchorążych Ave, 30-084 Krakow, Poland; marcin.piekarczyk@up.krakow.pl

<sup>2</sup> Department of Computer Physics and Quantum Informatics, Institute of Computer Science, Pedagogical University of Krakow, 2 Podchorążych Ave, 30-084 Krakow, Poland; lukasz.bibrzycki@up.krakow.pl

\* Correspondence: tomekhachaj@o2.pl

**Abstract:** In this paper, we describe the convolutional neural network (CNN)-based approach to the problems of categorization and artefact reduction of cosmic ray images obtained from CMOS sensors used in mobile phones. As artefacts, we understand all images that cannot be attributed to particles' passage through sensor but rather result from the deficiencies of the registration procedure. The proposed deep neural network is composed of a pretrained CNN and neural-network-based approximator, which models the uncertainty of image class assignment. The network was trained using a transfer learning approach with a mean squared error loss function. We evaluated our approach on a data set containing 2350 images labelled by five judges. The most accurate results were obtained using the VGG16 CNN architecture; the recognition rate (RR) was  $85.79\% \pm 2.24\%$  with a mean squared error (MSE) of  $0.03 \pm 0.00$ . After applying the proposed threshold scheme to eliminate less probable class assignments, we obtained a RR of  $96.95\% \pm 1.38\%$  for a threshold of 0.9, which left about  $62.60\% \pm 2.88\%$  of the overall data. Importantly, the research and results presented in this paper are part of the pioneering field of the application of citizen science in the recognition of cosmic rays and, to the best of our knowledge, this analysis is performed on the largest freely available cosmic ray hit dataset.

**Keywords:** cosmic rays; CMOS sensors; mobile phones; citizen science; deep neural network approximation; transfer learning; image processing



**Citation:** Hachaj, T.; Bibrzycki, Ł.; Piekarczyk, M. Recognition of Cosmic Ray Images Obtained from CMOS Sensors Used in Mobile Phones by Approximation of Uncertain Class Assignment with Deep Convolutional Neural Network. *Sensors* **2021**, *21*, 1963. <https://doi.org/10.3390/s21061963>

Academic Editor: Angel Diéguez

Received: 18 January 2021

Accepted: 8 March 2021

Published: 11 March 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



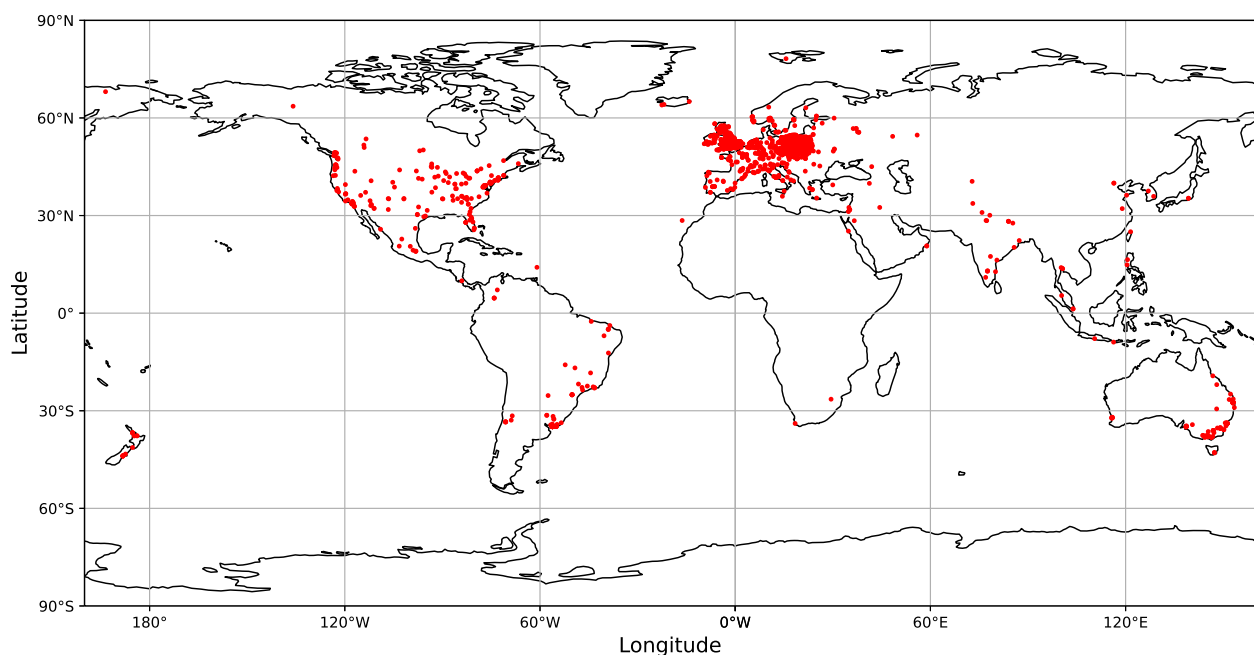
**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

In this paper, we describe a convolutional neural network (CNN)-based approach to the problems of categorization and artefact reduction of cosmic ray images obtained from CMOS sensors used in mobile phones. As artefacts, we understand all images that cannot be attributed to particles' passage through the sensor but rather result from the deficiencies of the registration procedure. Our approach is based on the morphological properties of particle tracks rather than their physical interpretation, although some studies [1–3] associated certain shapes of tracks like spots, wiggles (which we here call worms), etc., with muons, electrons, etc. Unambiguous mapping between track shapes and radiation types, however, requires detailed studies of radiation propagation in a sensor of given geometry. Such studies are challenging for commodity devices and, to the best of our knowledge, had not yet been performed. Therefore, we take an alternative approach and categorize the registered events relying solely on their morphology. This study is timely and opportune as it is compatible with any future interpretation of a given track in terms of particle species.

In 1912, Victor Hess conducted a series of balloon experiments, revealing that the electric conductivity of the atmosphere increases with the elevation above the sea level.

He boldly conjectured that the effect was due to the interaction of the atmosphere with the corpuscular charged particle radiation of extraterrestrial origin. More than 100 years after the discovery, due to their still undetermined origin, cosmic rays are being actively studied by astrophysicists. Further areas of interest include their implications for radiative safety [4], operation of electronic devices working both on Earth and in space [5,6], or even the earthquake prediction [7–9]. Of the several types of cosmic ray detectors [10,11], we focused on the semiconductor detectors [12]. Originally, they were conceived for measurements of particle energies, but with multi-sensor arrays equipped with many thousands of read-out channels and up to several hundred square meters of surface coverage, they became primarily used for particle tracking in nuclear and high-energy physics experiments [13]. Due to their low energy threshold, they also found applications beyond physical research, e.g., in medical imaging [14]. Notwithstanding their large number of applications, the basic physical processes upon which all semiconductor sensors are based are the same. Notably, the same physical processes are used in CMOS sensors applied in domestic electronic devices like video recorders or digital cameras used by mobile phones. The cameras of mobile phones are of particular interest for cosmic ray detection due to these devices' ubiquity and network connectivity. Combining these features makes the mobile phones an ideal framework for creating the global network of radiation detectors coupled to central data storage. This idea underpinned several particle detection initiatives like CRAYFIS [15–19], DECO [1,3,20,21], and CREDO [22,23]. The analysis presented in this paper is based on the CREDO detection data set, as this is currently the largest publicly available data set of particle images obtained with mobile phones. The range of the CREDO worldwide device infrastructure is shown in Figure 1. Currently, the total number of registering devices is over 10,000 and is increasing.



**Figure 1.** Locations of CREDO registered phone-based detectors.

### 1.1. State of the Art

As mentioned above, our goal was categorization and artefact rejection in cosmic ray images obtained from the CMOS sensors used in mobile phones by applying a two-dimensional analysis of the morphological properties to particle tracks. From the perspective of image processing and recognition, this problem should be solved by an algorithm from the group of algorithms devoted to the recognition of shapes and objects. Computer methods of shape feature extraction have been explored for many years. The most popular approaches are contour-based methods (i.e., Hausdorff distance, shape signature, boundary

moments, spectral transform, shape invariants, etc.) and region-based methods (i.e., invariant moments, shape matrices, convex hull, etc.) [24–26]. In the last years, object recognition has evolved from early methods that used hand-crafted representations and descriptions to state-of-the-art deep-learning-based approaches. Especially, convolutional neural networks have become one of the most successful image-based pattern recognition methods [27–30]. A transfer learning approach is among most useful techniques for adapting pre-trained CNN architectures to other image domains [31–34]. With the aid of transfer learning, it is possible to train an effective deep neural network (DNN) architecture with a limited number of training samples because it is possible to reuse previously trained kernels. DNN can also be successfully used in approximation tasks using uncertain data [34–36]. In practice, in some cases, it is possible to use the previously trained convolutional layers of a neural network as the input of a deep learning architecture. By using those pretrained layers, time and resources can be saved because rather than training from scratch, already available knowledge can be used.

### 1.2. Study Motivation

Conventional cosmic ray detectors range in scales from several centimeters square to about 3000 km square, like in the case of the Pierre Auger observatory [10]. Even such vast facilities must be considered of limited coverage, so to increase the number of registered showers, either the detector's surface should be increased or it should be run longer. Both options are economically prohibitive. So, the idea behind projects like CREDO is to trade the very limited coverage of a single phone sensor, which is of the order of a few millimeters square, for the huge number of particle-detecting devices scattered worldwide. This is an example of a citizen science project, where the research infrastructure is contributed by interested but not necessarily scholarly affiliated members.

However, the practical implementation of this attractive concept meets several difficulties that need to be properly considered. First, contrary to detectors working as parts of dedicated research infrastructures, the geometries, up and down times, and working conditions of individual sensors remain uncontrolled. Various devices' responses to similar particle signals may vary considerably depending on sensor geometry (height, width, and depth), noise level, and particular noise reduction algorithms implemented in the device (for a detailed discussion of sensor working conditions, see [23]). To enhance the participants' activity, the project relies on the gamification of measurements, with the adverse effect of the possibility of users cheating (i.e., deliberately producing artefacts). Thus, the scientific quality of a given device output generally needs to be evaluated by individual inspection, which is possible to only a limited extent, as currently there are over 18 million registered events and this number is expected to increase by two orders of magnitude [23]. The search for anomalies requires a flexible and adaptive approach.

Therefore, methods have to be developed for automatic artefact rejection as well as searching for particular signals of interest. In this context, the machine learning methods and convolutional neural networks are particularly suitable. Importantly, the research and results presented in this paper are in the pioneering field of the application of citizen science in the recognition of cosmic rays and, to the best of our knowledge, this analysis is performed on the largest freely available cosmic ray hit dataset.

From the perspective of motivation, the methods and specific tested architectures in our work are similar to those of [1] (project DECO). However, there are significant differences in image labeling for the classification purpose between our data set and that from DECO, which has convinced us that it is worth trying a different approach than the one proposed so far. According to [1], the class was also assigned by eye, by multiple people, and if humans disagreed on the classification, which occurred 10% of the time, the image was labeled as ambiguous and excluded from the training set. In our case, as can be seen in Table 1, about 66% of images were labeled unanimously by all judges. There might be two reasons for that: either the DECO data set is higher quality than ours or, more probably, a different labeling approach was undertaken; for example, in our case, judges

did not contact each other. How many judges participated in labeling the DECO data set was not specified. The large ambiguity in the data set is, in our opinion, cannot be ignored. Moreover, we can take advantage of it. Remember that uncertainties provide additional information about inter-class similarity.

**Table 1.** Distribution of votes for the certain class in the data set. Columns represent number of voices and rows represent certain classes. A judge could skip voting for a certain image if they were unsure as to which class it should be assigned.

	0	1	2	3	4	5
Spots	1790	103	82	34	80	261
Tracks	1832	136	91	73	109	109
Worms	1834	198	85	98	104	31
Artefacts	1115	82	3	0	0	1150

## 2. Materials and Methods

### 2.1. Problem Formulation

As mentioned above, it is currently not possible to associate unambiguously particular particle types with track morphologies. Therefore, we proceed in a general way and defined 3 morphological categories, which we dub spots, tracks, and worms, the latter being tracks with one or more wiggles of sufficiently large curvature for them to be visually distinguishable from tracks. The common feature of these 3 categories of signals is that they are quasi zero-dimensional (point-like) or one-dimensional (line-like). This is in line with the physical intuition that the microscopic objects colliding with the sensor's surface are able to deposit the charge within a small vicinity of the collision point. This entails point-like events if the particle hits the sensor at the angle close to  $90^\circ$  and line-like events if the particle hits the sensor at smaller angles. Additionally, we defined the artefact category that encompasses all events not satisfying the above requirements, i.e., those featuring large widths (being effectively two-dimensional) or related to too-large energy/charge deposit in the sensor. The approach that was undertaken to overcome the ambiguity of assigning images to a certain class was to ask a group of judges to assign each image to one of the four classes. Each judge could assign an image to only one class. They could also skip voting for certain images if unsure as to which class it should be assigned. According to this, if there are  $n$  judges, no more than  $n$  votes could be cast to a single class. It is also possible that a certain image would have zero votes cast on all classes. This situation occurs when all judges decide to skip voting this image when they are uncertain as to what class it belongs. We discuss the data set that was used in this experiment in Section 2.3. In summary, a labelled data set contains pairs: an RGB image  $I$  and a 4-dimensional vector of votes  $\bar{v}$ , each coordinate of which is the number of votes cast to a certain class.

The problem we aimed to solve was assigning a certain shape that is registered by the detector to one of the four classes: spots, tracks, worms, or artefacts. This is a classification problem, but we did not have ground truth image data labels defined as a crisp set. Due to the subjectivity of judges' decisions, it is possible that each image was assigned to more than one class. We could have filtered out all ambiguous data and leave only images that were unequivocally assigned to a single class; however this binary approach would have caused the loss of some important information about visual class similarities. Due to this, to model the uncertainty in judges' voting, we formulated this problem as an approximation rather than classification. Let  $I$  be an input image in the RGB color scale. To each image  $I$ , we want to assign a 4-dimensional real-valued vector with non-negative coordinates  $\bar{p}$ , which approximates the potential voting of judges, using a certain approximation function  $\Phi$ . Each dimension of the vector represents the number of votes that judges cast for a certain class.

$$\Phi(I) = \bar{p} \quad (1)$$

To make the approximation independent of the number of judges that participated in data set preparation, we also assumed that coordinates of vector  $\bar{p}$  are scaled in range  $[0, 1]$ , where 0 means that no judge voted for a certain class, while 1 indicates that all judges voted for it. We can easily transfer the votes of the judges from vector  $v$  to  $p$  by division of each coordinate of  $v$  by the number of judges  $k$ .

$$\bar{p} = \frac{\bar{v}}{k} \quad (2)$$

Vector  $\bar{p}$  is neither normalized nor do its coordinates sum to 1 intentionally. Finally, we have the following data set  $D$ :

$$D = \{(I_i, \bar{p}_i); i = [0, \dots, n]\}, \quad (3)$$

where  $I_i$  and  $\bar{p}_i$  are the  $i$ th image and the judges' labelling of the image, respectively;  $n$  is number of images in the data set.

## 2.2. Approximation of Uncertain Class Assignment with Deep Convolutional Neural Network

The data set in the form presented in Equation (3) can be easily adapted to a machine learning framework. As indicated in Section 1.1, the state-of-the-art approach for image embedding is the application of convolutional neural networks. We can either design a dedicated architecture that, after training, will generate valuable feature vectors, or use a pretrained model and retrain its non-convolutional layers using transfer learning. The first option requires a relatively large data set of example images, which might be difficult to manually label by judges. Because of this, we decided to use the second approach and adapt already trained network models. The second approach has a very important advantage: a pretrained convolutional network has many specialized filters that, in many cases, can be adapted to detect sophisticated objects (and shapes) in input images. The output of each CNN was processed by a global average pooling 2D layer and then propagated to the next layers. Because, as already mentioned in Section 2.1, we wanted to model an approximation rather than perform classification, we followed convolutional DNN in two layers: a dense (fully connected) layer with 128 neurons with ReLU activation function and the final dense layer with four neurons with a sigmoid activation function. A ReLU activation function is defined as [37]:

$$relu(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \quad (4)$$

A sigmoid neurons layer provides the opportunity for signal approximation. The schematic diagram of the system architecture is presented in Figure 2. The input dimension of the image was set to  $60 \times 60$  (see Section 2.3).

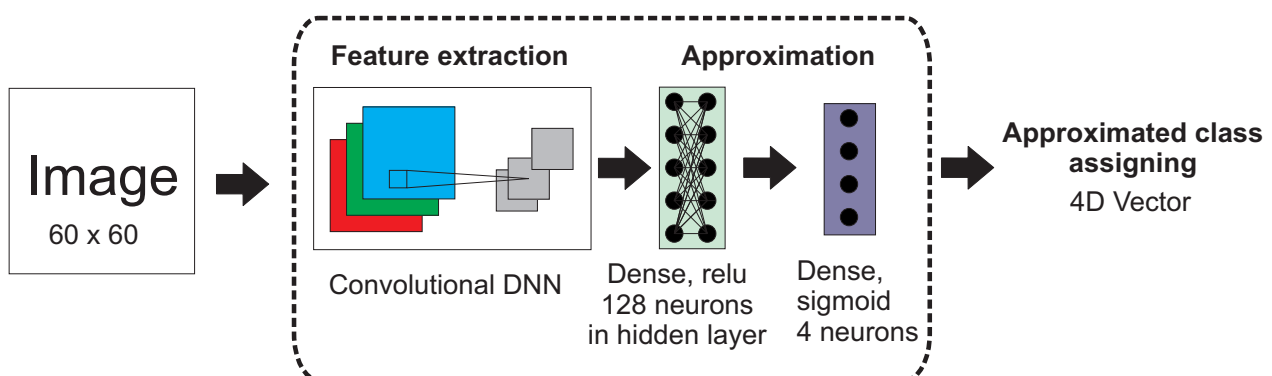


Figure 2. The architecture of the proposed deep convolutional neural network for uncertain class assigning.

The proposed approximator was trained using a first-order gradient-based Adam optimizer [38] with a mean squared error loss function; CNN layers weights remained fixed.

$$MSE = \frac{1}{n} \sum_0^n (\bar{p}_i - \bar{c}_i)^2, \quad (5)$$

where  $\bar{c}_i$  is the prediction returned by the network.

Several CNN-based feature extractors were considered, namely Xception [39], DenseNet201 [40], VGG16 [41], NASNetLarge [42], and MobileNetV2 [43]. Each network was pretrained on the ImageNet data set [44]. We chose a well-established and verified CNN model pretrained on various complex objects that are present in the ImageNet data set. The CNN architectures seem to be excessive for potentially fairly simple, highly processed images; however, the images were gathered by a large network of CMOS sensors that have nonuniform hardware and software parameters and they were not primarily designed as cosmic rays detectors. As such, although our data set contains 2350 images assigned to four classes by the judges, they are highly diverse, which is reflected by the ambiguous assessments of judges. As such, we decided to use embedding generated by general purpose pretrained CNN models that have convolutional multi-scale filters capable of modeling various possible typologies that might be registered by CMOS detectors. Our data set might not be large enough to train CNN-based embedding layers from scratch.

The cascade of convolutional filters with an architecture based on VGG16 was also used previously [1] and the authors decided to train it from scratch. As such, Winters et al. [1] had to undertake extensive data augmentation, which was not required in our case, because we adapted the VGG16 weights using transfer learning. As opposed to Winter et al. [1], we also applied basic image processing, which excluded salt-like noise from the input images.

The next problem that had to be addressed was assigning the class based on the certain result of voting  $\bar{p}_i$ . The most straightforward approach is to assign an image to the class that is represented by a coordinate of  $\bar{p}_i$ , which has the maximal value. If more than one coordinate has the same value, an image is assigned to a random class from those top-voted. This approach, however, could lead to situations where some images, for which approximation represents highly uncertainty of judges, will also be assigned to a class. For example, if there is the same distribution of votes to each class, the assignment will be random.

$$C_i = \max_{id} \bar{p}_i, \quad (6)$$

where  $C_i \in \{Dots, Lines, Worms, Artefacts\}$ .

For DNN-based approximation, it is hardly possible that two neurons generate an identical response; however, it is possible that a final layer will generate a vector with all coordinates being, for example, close to zero and simultaneously not much differing from each other. We intentionally did not apply a SoftMax activation in the last layer as in Winter et al. [1] because this approach is unsuitable for simulating (approximating) the voting of separate judges. A SoftMax activation function is defined as:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}, \quad (7)$$

where SoftMax is the exponent of the input  $x_i$  divided by a sum of the exponents of inputs  $x_j$  [37].

Instead of applying SoftMax, we preferred to use a threshold scheme with a border (threshold) parameter  $t$ . In this scheme, the image  $I_i$  is assigned to the class if and only if a maximal value of vector  $\bar{p}_i$  coordinate is greater than  $t$ :

$$C_i^t = \begin{cases} id & \text{if } \max_{id} \bar{p}_i > t \\ \emptyset & \text{if } \max_{id} \bar{p}_i \leq t' \end{cases} \quad (8)$$

where  $\emptyset$  means that the classifier left the object without assigning it to any class.

### 2.3. Image Data Set

As of October 2020, there were about 18 million events registered in the CREDO database from 16,000 devices scattered around the world. Of them, about 5 million of events meet the requirements allowing to qualify them as visible, which, among others, means that complete event metadata are recorded in the database and the integrated brightness (related to the energy deposit) falls below the fixed threshold [23]. Of the visible events, we selected the data set of 2350  $60 \times 60$  RGB images for this research. These images were subject to classification by 5 judges. After applying the class assignment method, 527 images were assigned to the spot class, 360 to the track class, 305 to the worm class, and 1158 to the artefact class.

The data set preparation procedure consisted of the following steps:

1. Selection of the subset of the trustworthy devices operating in controlled conditions;
2. Taking the image sample from trustworthy devices containing all morphologies of interest;
3. Assigning the dataset elements to four classes with the help of 5 judges with the majority vote while retaining the number of votes cast for each class.

As there were potentially a few sources of artefacts like hardware malfunction, insufficiently tight lens covering, or outright user cheating, we decided to introduce the notion of trustworthy devices. These are devices that performed the experiment in controlled conditions. To create a representative dataset for this article, we used data from our own devices that were run and operated under the supervision of CREDO researchers. We used the signals only from those devices so that the possibility of using cheating-affected data was entirely eliminated. Table 1 presents the distribution of votes for the classes in the data set.

### 2.4. Image Preprocessing

Before the image is processed by the CNN, some initial preprocessing is performed. The goal of preprocessing is to remove all objects but the signal of interest from the image set. The signal of interest is defined as white objects with sufficiently high color value in the RGB space. Preprocessing is performed with the following image processing steps (Figure 3):

1. Let  $I$  be an input image in the RGB color scale (Figure 3A). First, the image is converted to gray scale. The gray value is calculated as the linear combination of the weighted RGB channels values by a standard OpenCV 4.2.0.32 function (see details in source code).

$$I_g = \text{gray}(I)$$

2. An object of interest is detected by maximizing the separability of the resultant classes in gray levels using an Otsu algorithm [45] (Figure 3B). The result is stored in binary mask.

$$M_{ask} = \text{Otsu}(I_g)$$

3. The binary image  $M_{ask}$  is dilated and then opened using image morphology operations [46] with an elliptical kernel with a diameter of 5 pixels. After this operation, the objects detected by the Otsu algorithm have slightly increased their borders and nearby objects are merged together. Opening also removes small holes in regions (Figure 3C).

$$M_{ask1} = \text{Dilate}(M_{ask}, \text{kernel})$$

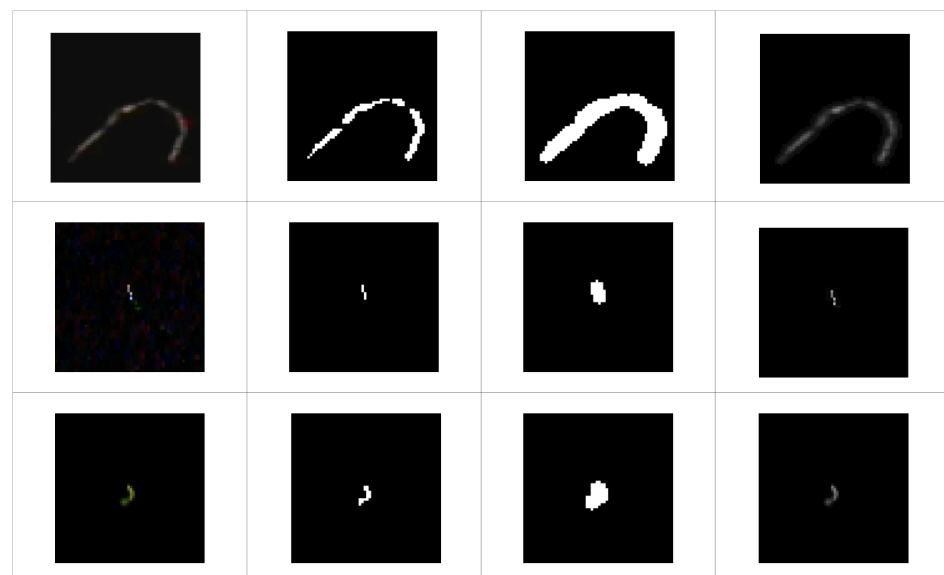
$$M_{ask2} = \text{Open}(M_{ask1}, \text{kernel})$$

- The final image  $I_p$  is generated by extracting from the gray scale image  $I_g$  only those pixels that are in the non-zero region of the binary mask  $M_{ask2}$ . The rest of the pixels in  $I_g$  are set to zero (Figure 3D).

$$I_{out} = I_g \& M_{ask2}$$

The above pipeline is repeated for each image  $I_k$  from the data set described in Section 2.3. The set of output images  $I_{out_k}$  is presented as an input image to the CNN. The role of the above image processing pipeline is to mime the procedure that is performed by each judge, who assigned images to a certain class. Judges only considered the curvatures of the object; the backgrounds were irrelevant to them. The proposed algorithm generates a binary mask whose role is to enhance only the object detected by the Otsu algorithm and the small surroundings of those objects, because the borders of those regions are blurred. We chose a kernel with very small diameter (5), which has the potential to fill holes with a diameter of about 3 pixels and to remove salt-like artefacts. Due to this small kernel diameter, the curvature of the detected objects remains the same. Perhaps it is possible to skip the above data processing; however, all background noises will be present in CNN embedding, which will disturb the final recognition process.

(A) Input image (B) Otsu algorithm (C) Binary mask (D) Output image



**Figure 3.** Image preprocessing pipeline. Each row represents a separate image. Each column shows the final result of a single preprocessing transformation performed on the image. The following processing steps are included: input image loading with conversion to grayscale format (A), automatic thresholding via the Otsu algorithm (B), creation of a binary mask (C), and definitive filtering overlaying the binary mask on the grayscale image (D).

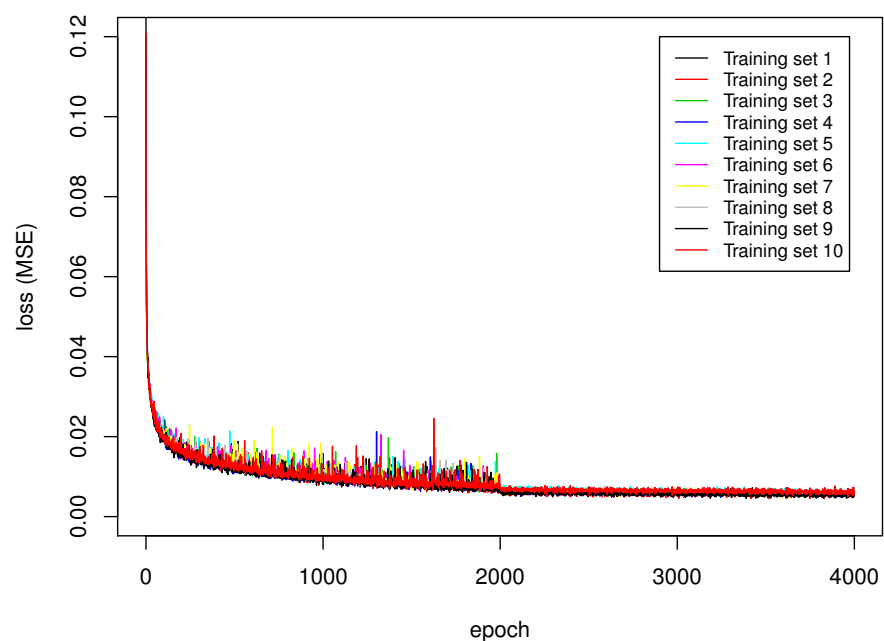
### 3. Results

The proposed image preprocessing and neural network approximation pipeline introduced in Sections 2.2 and 2.4 were evaluated on the data set discussed in Section 2.3. The solution was implemented in Python 3.6. Among most important packages that were used were Tensorflow 2.1 for machine learning, deep neural networks Keras 2.3.1 library, and OpenCV-python 4.2.0.32 for image processing. Additional data evaluation was conducted in R version 3.6.2. The research was computed on a PC with an Intel i7-9700F 3.00 GHz CP, 64 GB RAM, NVIDIA GeForce RTX 2060 GPU, and operating on Windows 10 OS. Both source codes and data are available for download from an online repository ([https://github.com/browarsoftware/credo\\_dnn\\_rays\\_recognition](https://github.com/browarsoftware/credo_dnn_rays_recognition), accessed on 10 March 2021).



The training parameters were set to 4000 training epochs and batch size to 64. The learning rate for the first 2000 iterations was 0.001 and for the next 2000 was 0.0001. The learning rate governs the step size of the gradient descent method (see parameter  $\alpha$  in [38]). The data set was split into a training data set that contained 90% of the objects (2115 images) and a validation data set with 10% of the objects (235 images). Each network with different CNN feature extractors was evaluated 10 times on different random data sets. Each training data set had 2115 elements randomly chosen from the 2350 images (without replacement); the remaining 235 images were assigned to the validation data set. In case of tied-voting by the judges in Equation (6), we did not re-randomize classes assigning for those ten sets. The results were averaged and the numbers in all tables are percentage values.

Table 2 presents the recognition rate and mean square error of networks with various input convolutional architectures. The recognition rate is the total number of correctly identified images from the validation data set divided by the total number of images in the validation data set [47]. The highest recognition rate was obtained using VGG16. The second highest recognition rate for DenseNet201 differed only by 1.1% and had slightly smaller variance. Both networks have the smallest mean square error (MSE). During the training of all networks, the loss (MSE) function was minimized until reaching a certain value, which depends on the input CNN, the initial random weights choice, and the training data set (Figure 4). The relatively low variance of the values in Table 2 indicates, however, that the choice of CNN has the strongest impact on the overall results, and the network effectiveness is robust to initial random parameters and training data set choice.



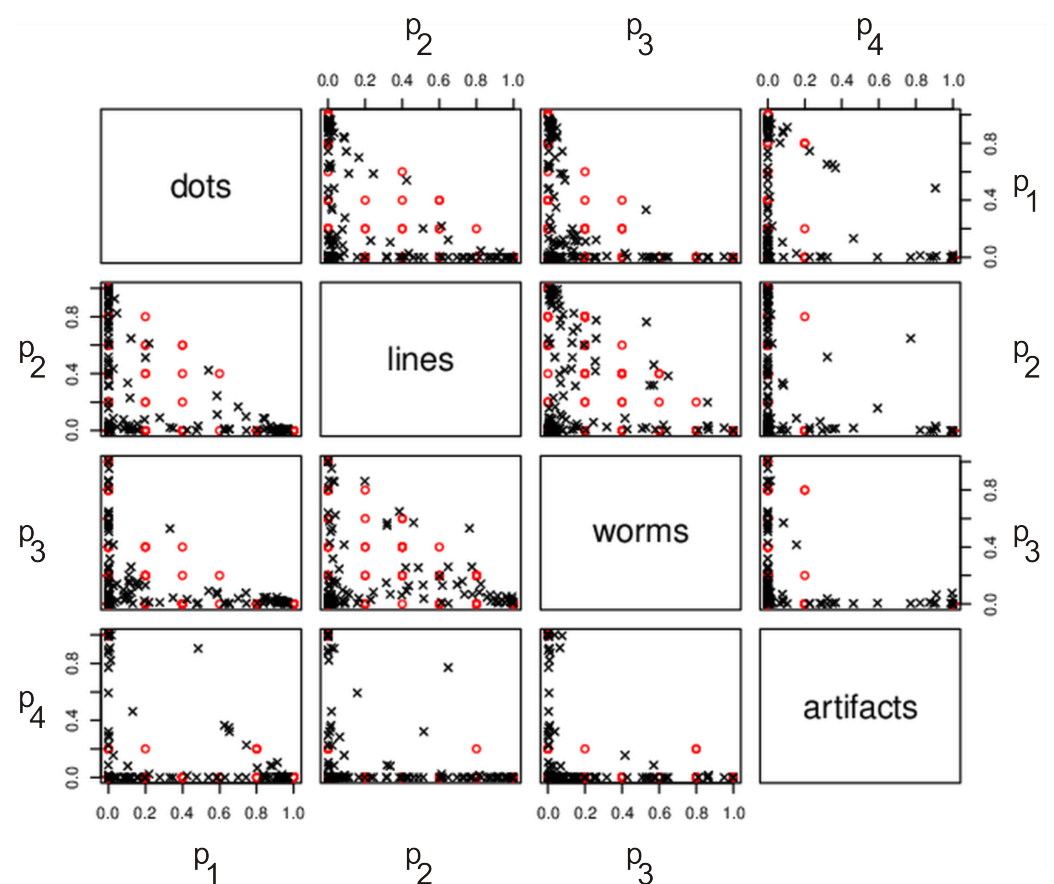
**Figure 4.** Changes in the loss (mean square error (MSE)) function during the training of the neural network with the VGG16 input layer. Time is measured in epochs. Each line represents a different random training set. After 2000 epochs, we observed smaller variance in the loss caused by reducing the learning rate from 0.001 to 0.0001.

**Table 2.** Recognition rate and mean square error of networks with various input convolutional architectures.

Input Convolutional Architecture	Recognition Rate	Mean Squared Error
VGG16	85.79 $\pm$ 2.24	0.03 $\pm$ 0.00
NASNetLarge	81.66 $\pm$ 2.53	0.04 $\pm$ 0.01
MobileNetV2	78.43 $\pm$ 2.06	0.05 $\pm$ 0.01
Xception	81.49 $\pm$ 2.94	0.04 $\pm$ 0.01
DenseNet201	84.68 $\pm$ 1.77	0.03 $\pm$ 0.00

Figure 5 presents a pairs plot showing the bivariate relationships between all pairs of variables for one of the validation data sets. Red dots are judge-labeled while black crosses are predicted values. Predictions were performed using a neural network with the VGG16 input layer. As observed, judge-labeled values are obviously discreet; because of that, most values overlap and are represented by the same points in space.

Tables 3–7 present the confusion matrices of the networks with input convolutional architectures VGG16, NASNetLarge, MobileNetV2, Xception, and DenseNet201, respectively. Matrices are row-normalized and each row represents a judge label. Columns represent the predicted label. In all cases, over 93% of artefacts were correctly classified. The true positive rate of the rest of the classes depended on type of input convolutional neural network. The highest recognition rates for spots, tracks, and worms were obtained using VGG16. The highest recognition rate for artefacts was obtained using the DenseNet201 architecture; however, the difference between this network and VGG16 was only 0.68% with similar variance values.



**Figure 5.** The pairs plot showing the bivariate relationships between all pairs of variables in one of the validation data sets. The pairs plot is represented as scatterplots between all pairs of these variables. In the first line, there is a scatter plot of spots and tracks, then one of spots and worms, and then one of spots and artefacts. The second row presents tracks and spots (symmetric to the first), tracks and worms, and so on. For a detailed description of the pairs plot, see [48]. Red dots are judge-labeled while black crosses are predicted values. Predictions were performed using a neural network with the VGG16 input layer. Values on the axis are the coordinates of vector  $\bar{p}$  (see Equation (2)). For example,  $p_1 = 1$  means that all judges voted for dot and  $p_2 = 0.5$  means that half of the judges voted for line.

**Table 3.** Confusion matrix of the network with input convolutional VGG16 architecture.

	Spots	Tracks	Worms	Artefacts
Spots	$90.84 \pm 4.16$	$1.38 \pm 1.88$	$5.04 \pm 2.82$	$2.74 \pm 1.78$
Tracks	$7.46 \pm 3.80$	$73.31 \pm 11.50$	$15.10 \pm 9.01$	$4.14 \pm 2.53$
Worms	$6.14 \pm 6.01$	$22.64 \pm 10.70$	$62.59 \pm 9.92$	$8.64 \pm 4.19$
Artefacts	$2.54 \pm 1.23$	$0.80 \pm 1.05$	$2.70 \pm 1.60$	$93.97 \pm 1.93$

**Table 4.** Confusion matrix of the network with the input convolutional NASNetLarge architecture.

	Spots	Tracks	Worms	Artefacts
Spots	$89.52 \pm 5.40$	$3.31 \pm 3.13$	$3.76 \pm 3.29$	$3.41 \pm 2.05$
Tracks	$9.27 \pm 4.13$	$62.26 \pm 5.06$	$17.96 \pm 4.93$	$10.52 \pm 5.46$
Worms	$7.83 \pm 4.16$	$26.41 \pm 9.08$	$51.66 \pm 6.54$	$14.11 \pm 5.30$
Artefacts	$2.44 \pm 1.79$	$1.58 \pm 0.39$	$2.71 \pm 1.50$	$93.27 \pm 2.71$

**Table 5.** Confusion matrix of the network with the input convolutional MobileNetV2 architecture.

	Spots	Tracks	Worms	Artefacts
Spots	$78.66 \pm 3.46$	$13.37 \pm 4.03$	$4.34 \pm 2.70$	$3.63 \pm 2.34$
Tracks	$12.28 \pm 6.03$	$56.14 \pm 5.78$	$24.40 \pm 4.50$	$7.18 \pm 5.96$
Worms	$7.15 \pm 5.44$	$25.63 \pm 9.40$	$50.84 \pm 7.89$	$16.39 \pm 3.87$
Artefacts	$2.09 \pm 1.29$	$1.22 \pm 0.61$	$3.15 \pm 2.04$	$93.54 \pm 1.74$

**Table 6.** Confusion matrix of network with input convolutional Xception architecture.

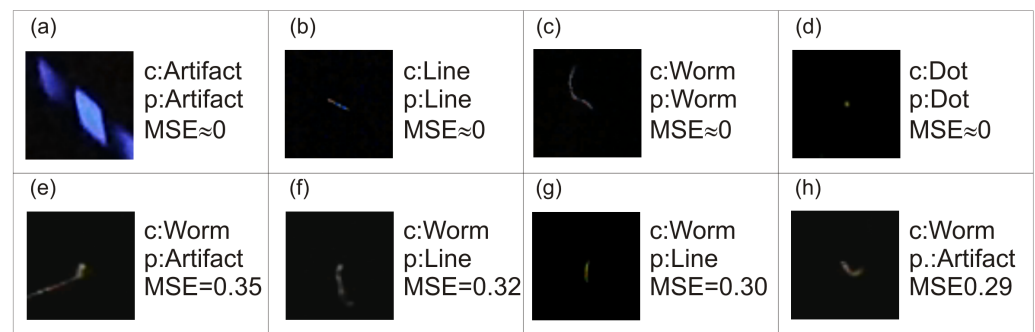
	Spots	Tracks	Worms	Artefacts
Spots	$84.29 \pm 4.04$	$3.69 \pm 2.83$	$6.49 \pm 2.86$	$5.53 \pm 3.67$
Tracks	$8.22 \pm 5.66$	$60.21 \pm 7.30$	$23.58 \pm 7.44$	$7.99 \pm 4.66$
Worms	$7.58 \pm 4.77$	$21.18 \pm 5.23$	$60.18 \pm 8.70$	$11.06 \pm 4.65$
Artefacts	$2.01 \pm 1.21$	$1.40 \pm 1.09$	$3.14 \pm 1.71$	$93.45 \pm 2.18$

**Table 7.** Confusion matrix of the network with the input convolutional DenseNet201 architecture.

	Spots	Tracks	Worms	Artefacts
Spots	$87.44 \pm 4.97$	$4.79 \pm 2.74$	$4.25 \pm 2.61$	$3.52 \pm 2.54$
Tracks	$5.11 \pm 3.06$	$71.03 \pm 5.43$	$20.33 \pm 5.41$	$3.53 \pm 3.49$
Worms	$6.03 \pm 5.00$	$23.46 \pm 7.71$	$61.90 \pm 10.16$	$8.61 \pm 3.15$
Artefacts	$2.63 \pm 0.86$	$0.79 \pm 0.87$	$1.93 \pm 1.17$	$94.65 \pm 1.86$

Figure 6 visualizes an example of the best and worse approximations for predictions performed using the neural network with the VGG16 input layer.

Tables 8–10 present confusion matrices after applying the threshold scheme (8) with various thresholds to the network with the VGG16 features generator. Only VGG16 was evaluated because it proved to be the most reliable in previous experiments. The threshold scheme eliminates less certain predictions with a threshold of  $t$ . The table captions provide information about threshold  $t$ , validation data that remain after applying the threshold, data that remain after being split into classes, and overall recognition rate.



**Figure 6.** Example of best and worse approximations. C means correct (judge labeled) class, p is predicted class, and MSE is a mean squared error between judge-labeled value and predicted value. Predictions were performed using a neural network with the VGG16 input layer. The MSE of the images in the first row is below 0.005. The images (a–d) represent correct assignments, while (e–h) illustrate cases of misclassification.

**Table 8.** Confusion matrix of network with input VGG16 and threshold scheme (8)  $t = 0.50$ . Data remaining =  $85.36 \pm 1.79$  (S 79.92%  $\pm$  4.37%; T  $80.03 \pm 3.40$ ; W  $56.98 \pm 16.92$ ; A  $97.89 \pm 1.41$ ), recognition rate =  $92.11 \pm 1.92$ .

	Spots (S)	Tracks (T)	Worms (W)	Artefacts (A)
Spots	$94.00 \pm 3.87$	$2.14 \pm 1.90$	$1.40 \pm 2.13$	$2.46 \pm 2.47$
Tracks	$1.99 \pm 2.14$	$82.64 \pm 5.11$	$14.07 \pm 4.40$	$1.30 \pm 2.38$
Worms	$2.65 \pm 5.33$	$17.09 \pm 10.25$	$75.87 \pm 13.51$	$4.39 \pm 4.15$
Artefacts	$1.01 \pm 1.12$	$1.01 \pm 0.69$	$1.86 \pm 0.86$	$96.12 \pm 1.27$

**Table 9.** Confusion matrix of the network with input VGG16 and threshold schema (8)  $t = 0.75$ . Data remaining =  $74.26 \pm 2.56$  (S 63.12%  $\pm$  4.63%; T  $55.60 \pm 5.83$ ; W  $34.71 \pm 7.93$ ; A  $96.46 \pm 2.08$ ), recognition rate =  $94.95 \pm 1.00$ .

	Spots (S)	Tracks (T)	Worms (W)	Artefacts (A)
Spots	$96.23 \pm 2.29$	$0.92 \pm 1.49$	$0.36 \pm 1.13$	$2.49 \pm 2.38$
Tracks	$0.83 \pm 2.64$	$88.96 \pm 6.79$	$8.72 \pm 5.25$	$1.49 \pm 3.46$
Worms	$1.00 \pm 3.16$	$14.24 \pm 11.74$	$79.54 \pm 10.97$	$5.22 \pm 5.75$
Artefacts	$0.65 \pm 0.64$	$0.56 \pm 0.48$	$1.80 \pm 0.81$	$97.00 \pm 0.95$

**Table 10.** Confusion matrix of the network with input VGG16 and threshold schema (8)  $t = 0.90$ . Data remaining =  $62.60 \pm 2.88$  (S 42.03  $\pm$  5.65; T  $29.91 \pm 6.55$ ; W  $18.445 \pm 7.22$ ; A  $94.97 \pm 2.17$ ), recognition rate =  $96.95 \pm 1.38$ .

	Spots (S)	Tracks (T)	Worms (W)	Artefacts (A)
Spots	$98.71 \pm 2.99$	$0.45 \pm 1.44$	$0.00 \pm 0.00$	$0.84 \pm 1.78$
Tracks	$1.43 \pm 4.52$	$88.89 \pm 12.78$	$8.85 \pm 8.78$	$0.83 \pm 2.64$
Worms	$0.00 \pm 0.00$	$7.43 \pm 15.82$	$89.65 \pm 15.52$	$2.92 \pm 6.23$
Artefacts	$.38 \pm 0.50$	$0.27 \pm 0.44$	$1.64 \pm 0.81$	$97.70 \pm 1.24$

#### 4. Discussion

As shown in Section 3, the proposed deep convolutional neural network architecture is capable of approximating uncertain class assignments that were performed manually by a group of judges. There are two measures we used to evaluate our solution: RR and MSE. Although there are a large number of trainable parameters in classification layers, the high recognition rate evaluated in 10-fold cross-validation assures that the network

was not overtrained and still has generalization ability. All convolutional feature extractors have relatively small MSE, while VGG16 and DenseNet201 seem to be the best for the task. The value of MSE corresponds to the recognition rate of the network: the smaller the MSE, the better the recognition rate of the network. This is an important finding because it indicates that the uncertainty modelling of judges' decisions was correctly designed (Table 2). The training of the proposed architecture is stable and follows expectations. The lowering of the learning rate value stabilizes the variation in the loss functions and slightly decreases the MSE (Figure 4). Lowering the learning rate after a certain number of iterations of the gradient-descent method lowers the influence of the gradient on the final solution. This allows for a better adjustment of the solution to the local minimum. According to confusion matrices presented in Tables 3–7, the artefact class was the easiest to recognize. This is probably because those images differ the most from other classes despite artefacts potentially having various forms. The second easiest to classify object was spots because spots are among the best-defined potential shapes that can be found in the data set. The next two classes, track and worm, were more problematic. These two classes are most often confused with each other due to the subjectivity of the judgement of specialists assigning images to those two classes. In case of the network using the VGG16 feature extractor, nearly  $15.10\% \pm 9.01\%$  of tracks were incorrectly assigned to the worm class, while  $22.64\% \pm 10.70\%$  of worms were incorrectly assigned as tracks. As shown in Figure 6, the difference between tracks and worms is very subjective: there is not much visible difference between a track (Figure 6b) and a worm (Figure 6g). It was difficult to guess the judges' reasoning in this case. Worms were confused with artefacts: in case of VGG16, incorrect classification between those classes was  $8.64\% \pm 4.19\%$ . This situation was also caused by judges' subjectivity. Due to the MSE being quite low, the proposed architecture correctly models the judges' decision despite there only being five judges and the shape of the worm class was not clearly defined (see Section 2.3). There are two possible solutions to overcome this problem. The first is to increase the number of judges and to define each class more precisely; however, this does not guarantee improving the true positive rate of worm and track classes. The second possibility is to apply the threshold scheme (8). Application of this scheme involves a trade-off between the accuracy and the number of objects that can be classified. As shown in Tables 8–10, even the application of the lowest considered threshold  $t = 0.50$  improves the true positive rate of all classes (compare with Table 3). For example, the true positive rate of the worm classes improved from  $62.59\% \pm 9.9\%$  to  $75.87\% \pm 13.51\%$  when  $t = 0.50$ , to  $79.54\% \pm 10.97\%$  when  $t = 0.75$ , and to  $89.65\% \pm 15.52\%$  when  $t = 0.90$ . This operation, however, results in  $56.98\% \pm 16.92\%$ ,  $34.71\% \pm 7.93\%$ , and  $18.445\% \pm 7.22\%$  of worms being appropriately classified, respectively. Due to this finding, threshold  $t$  has to be chosen carefully, considering many factors of certain detection. At this moment, it is difficult to compare our results directly with those from Winter et al. [1], mainly because the DECO dataset is not publicly available; however, the accuracy of the results we obtained is very similar to those previously reported: spots 98.9% (our result from Table 10 is 98.7%), tracks 95.4% (ours: 88.9%), worms 92.9% (ours: 90.0%), and artefacts 98.2% (ours: 97.7%). Notably, we did not exclude any object either from the training or validation dataset due to labeling disagreement between judges, as was performed for the DECO dataset. Certainly, the image quality and the labelling process of the dataset have considerable impacts on the results of a method. In our case, we used approximation rather than a classification approach in DNN training, which seems to be reasonable with the presence of uncertainty in class assigning. Based on our experience, we think that unless some standardized approach to class assigning is established, uncertainties are inevitable. Therefore, the classification model should not only be able to deal with them but also take advantages of them, as does our proposed method.

## 5. Conclusions

Based on the research presented in this paper, we conclude that the proposed recognition algorithm based on the approximation of uncertain class assignment with a deep

convolutional neural network together with threshold scheme seems to be promising method to identify various classes of cosmic ray images obtained from CMOS sensors used in mobile phones. We recommend using VGG16 as the feature extractor. The performance of our method using VGG16 is not considerably different from other CNN networks beside MobileNetV2. According to Table 2, both VGG16 and DenseNet201 have the smallest mean squared error; however, DenseNet201 has a more complex architecture that affects its performance. Increasing the complexity and depth of artificial neural networks for classification is not always necessary to achieve state-of-the-art results [49]. The appropriate choice of threshold  $t$  highly depends on the detection setup, because it is a trade-off between the accuracy and number of objects that can be classified. Because the proposed approach is based on machine learning, a high-quality training data set is a crucial component to obtain reliable classification. To improve the obtained results, a larger data set of images that contains more objects labelled by a larger number of scientists must be created. Moreover, we think that VGG16 might be a too-extensive architecture for features extraction. After acquiring the larger data set that we mentioned above, research should be conducted to optimize the CNN to reduce the number of layers and weights. A smaller CNN architecture will result in the acceleration of training and computation speed and will make the model more portable by limiting the amount of required memory to store all its parameters.

**Author Contributions:** T.H. was responsible for conceptualization, proposed methodology, software implementation, and writing of original draft; L.B. was responsible for data preparation and writing the original draft; M.P. was responsible for data presentation, writing review, editing, and formal analysis. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded under the Pedagogical University of Krakow statutory research grant, which was funded by subsidies for science granted by the Polish Ministry of Science and Higher Education.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Both source codes and data are available for download from the online repository [https://github.com/browarsoftware/credo\\_dnn\\_rays\\_recognition](https://github.com/browarsoftware/credo_dnn_rays_recognition), accessed on 10 March 2021.

**Acknowledgments:** Authors express their gratitude for the opportunity to use the data obtained with the CREDO infrastructure. We also thank the CREDO-ML group for providing us with the human-preclassified subset of CREDO image data, which we used in our research. In particular, thanks are due to the following members of the group: Michał Niedźwiedzki, Sławomir Stuglik, Krzysztof Rzecki, and Olaf Bar.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Winter, M.; Bourbeau, J.; Bravo, S.; Campos, F.; Meehan, M.; Peacock, J.; Ruggles, T.; Schneider, C.; Simons, A.L.; Vandenbroucke, J. Particle identification in camera image sensors using computer vision. *Astropart. Phys.* **2019**, *104*, 42–53. [CrossRef]
2. Groom, D. Cosmic Rays and Other Nonsense in Astronomical CCD Imagers. In *Scientific Detectors for Astronomy*; Amico, P., Beletic, J.W., Beletic, J.E., Eds.; Astrophysics and Space Science Library; Springer: Dordrecht, The Netherlands, 2004; Volume 300. [CrossRef]
3. Vandenbroucke, J.; BenZvi, S.; Bravo, S.; Jensen, K.; Karn, P.; Meehan, M.; Peacock, J.; Plewa, M.; Ruggles, T.; Santander, M.; et al. Measurement of cosmic-ray muons with the Distributed Electronic Cosmic-ray Observatory, a network of smartphones. *J. Instrum.* **2016**, *11*. [CrossRef]
4. Shea, M.A.; Smart, D.F. Cosmic Ray Implications for Human Health. In *Cosmic Rays and Earth*; Bieber, J.W., Eroshenko, E., Evenson, P., Flückiger, E.O., Kallenbach, R., Eds.; Springer: Amsterdam, The Netherlands, 2000; pp. 187–205.
5. Kim, J.; Lee, J.; Han, J.; Meyyappan, M. Caution: Abnormal Variability Due to Terrestrial Cosmic Rays in Scaled-Down FinFETs. *IEEE Trans. Electron Devices* **2019**, *66*, 1887–1891. [CrossRef]
6. Höeffgen, S.K.; Metzger, S.; Steffens, M. Investigating the Effects of Cosmic Rays on Space Electronics. *Front. Phys.* **2020**, *8*, 318. [CrossRef]

7. Foppiano, A.J.; Ovalle, E.M.; Bataille, K.; Stepanova, M. Ionospheric evidence of the May 1960 earthquake Concepción? *Geofis. Int.* **2008**, *47*, 179–183.
8. Romanova, N.V.; Pilipenko, V.A.; Stepanova, M.V. On the magnetic precursor of the Chilean earthquake of February 27, 2010. *Geomagn. Aeron.* **2015**, *55*, 219–222. [[CrossRef](#)]
9. He, L.; Heki, K. Three-Dimensional Tomography of Ionospheric Anomalies Immediately Before the 2015 Illapel Earthquake, Central Chile. *J. Geophys. Res. (Space Phys.)* **2018**, *123*, 4015–4025. [[CrossRef](#)]
10. Pierre Auger Collaboration. The Pierre Auger Cosmic Ray Observatory. *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.* **2015**, *768*, 172–213. [[CrossRef](#)]
11. Ahlers, M.; Halzen, F. Opening a new window onto the universe with IceCube. *Prog. Part. Nucl. Phys.* **2018**, *102*, 73–88. [[CrossRef](#)]
12. Spieler, H. *Semiconductor Detector Systems*; Clarendon Press: Oxford, UK, 2005. [[CrossRef](#)]
13. Szumlak, T. Silicon detectors for the LHC Phase-II upgrade and beyond RD50 Status report. *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.* **2020**, *958*, 162187. [[CrossRef](#)]
14. Ruat, M.; d’Aillon, E.G.; Verger, L. 3D semiconductor radiation detectors for medical imaging: Simulation and design. In Proceedings of the 2008 IEEE Nuclear Science Symposium Conference Record, Dresden, Germany, 19–25 October 2008; pp. 434–439. [[CrossRef](#)]
15. Unger, M.; Farrar, G. (In)Feasibility of Studying Ultra-High-Energy Cosmic Rays with Smartphones. *arXiv* **2015**, arXiv:1505.04777.
16. Kumar, R. Tracking Cosmic Rays by CRAYFIS (Cosmic Rays Found in Smartphones) Global Detector. In Proceedings of the 34th International Cosmic Ray Conference (ICRC 2015), Hague, The Netherlands, 30 July–6 August 2015; p. 1234. [[CrossRef](#)]
17. Borisyak, M.; Usvyatsov, M.; Mulhearn, M.; Shimmin, C.; Ustyuzhanin, A. Muon Trigger for Mobile Phones. In Proceedings of the 22nd International Conference on Computing in High Energy and Nuclear Physics (CHEP2016), San Francisco, CA, USA, 14–16 October 2016; Volume 898, p. 032048. [[CrossRef](#)]
18. Albin, E.; Whiteson, D. Feasibility of Correlated Extensive Air Shower Detection with a Distributed Cosmic Ray Network. *arXiv* **2021**, arXiv:2102.03466.
19. Whiteson, D.; Mulhearn, M.; Shimmin, C.; Cranmer, K.; Brodie, K.; Burns, D. Searching for ultra-high energy cosmic rays with smartphones. *Astropart. Phys.* **2016**, *79*, 1–9. [[CrossRef](#)]
20. Vandenbroucke, J.; Bravo, S.; Karn, P.; Meehan, M.; Peacock, J.; Plewa, M.; Ruggles, T.; Schultz, D.; Simons, A.L. Detecting particles with cell phones: The Distributed Electronic Cosmic-ray Observatory. *arXiv* **2015**, arXiv:1510.07665.
21. Meehan, M.; Bravo, S.; Campos, F.; Peacock, J.; Ruggles, T.; Schneider, C.; Simons, A.L.; Vandenbroucke, J.; Winter, M. The particle detector in your pocket: The Distributed Electronic Cosmic-ray Observatory. *arXiv* **2017**, arXiv:1708.01281.
22. Homola, P.; Beznosko, D.; Bhatta, G.; Bibrzycki, Ł.; Borczyńska, M.; Bratek, Ł.; Budnev, N.; Burakowski, D.; Alvarez-Castillo, D.E.; Cheminant, K.A.; et al. Cosmic-Ray Extremely Distributed Observatory. *Symmetry* **2020**, *12*, 1835. [[CrossRef](#)]
23. Bibrzycki, Ł.; Burakowski, D.; Homola, P.; Piekarczyk, M.; Niedźwiecki, M.; Rzecki, K.; Stuglik, S.; Tursunov, A.; Hnatyk, B.; Castillo, D.E.; et al. Towards A Global Cosmic Ray Sensor Network: CREDO Detector as the First Open-Source Mobile Application Enabling Detection of Penetrating Radiation. *Symmetry* **2020**, *12*, 1802. [[CrossRef](#)]
24. Sahibuddin, S.; Sjarif, N.N.A.; Hussein, I.S. Shape feature extraction methods based pattern recognition: a survey. *Open Int. J. Inform.* **2018**, *6*, 26–41.
25. Yadav, P.; Yadav, B. A survey on application of image processing techniques in object shape recognition. *Int. J. Electron. Eng.* **2018**, *10*, 157–164.
26. Narottambhai, M.; Tandel, P. A Survey on Feature Extraction Techniques for Shape based Object Recognition. *Int. J. Comput. Appl.* **2016**, *137*, 16–20. [[CrossRef](#)]
27. Gao, M.; Jiang, J.; Zou, G.; John, V.; Liu, Z. RGB-D-Based Object Recognition Using Multimodal Convolutional Neural Networks: A Survey. *IEEE Access* **2019**, *7*, 43110–43136. [[CrossRef](#)]
28. Kaya, M.; Bilge, H. Deep Metric Learning: A Survey. *Symmetry* **2019**, *11*, 1066. [[CrossRef](#)]
29. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A Survey of Deep Learning-Based Object Detection. *IEEE Access* **2019**, *7*, 128837–128868. [[CrossRef](#)]
30. Loncomilla, P. Object Recognition using Local Invariant Features for Robotic Applications: A Survey. *Pattern Recognit.* **2016**, *60*. [[CrossRef](#)]
31. Krishna, S.; Kalluri, H. Deep learning and transfer learning approaches for image classification. *IEEE Trans. Robot.* **2019**, *7*, 427–432.
32. Falco, P.; Lu, S.; Natale, C.; Pirozzi, S.; Lee, D. A Transfer Learning Approach to Cross-Modal Object Recognition: From Visual Observation to Robotic Haptic Exploration. *IEEE Trans. Robot.* **2019**, *35*, 987–998. [[CrossRef](#)]
33. Wang, S.; Zhang, L.; Zuo, W.; Zhang, B. Class-Specific Reconstruction Transfer Learning for Visual Recognition Across Domains. *IEEE Trans. Image Process.* **2020**, *29*, 2424–2438. [[CrossRef](#)]
34. Perera, P.; Patel, V.M. Deep Transfer Learning for Multiple Class Novelty Detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 11536–11544. [[CrossRef](#)]

35. Daw, A.; Thomas, R.; Carey, C.; Read, J.; Appling, A.; Karpatne, A. Physics-Guided Architecture (PGA) of Neural Networks for Quantifying Uncertainty in Lake Temperature Modeling. In Proceedings of the 2020 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, Cincinnati, OH, USA, 7–9 May 2020; pp. 532–540. [[CrossRef](#)]
36. Hachaj, T.; Miazga, J. Image Hashtag Recommendations Using a Voting Deep Neural Network and Associative Rules Mining Approach. *Entropy* **2020**, *22*, 1351. [[CrossRef](#)]
37. Heaton, J.; Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; The MIT Press: Cambridge, UK, 2016; ISBN 0262035618. [[CrossRef](#)]
38. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations—ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
39. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.
40. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269.
41. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:abs/1409.1556.
42. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning Transferable Architectures for Scalable Image Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8697–8710.
43. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
44. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis. (IJCV)* **2015**, *115*, 211–252. [[CrossRef](#)]
45. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [[CrossRef](#)]
46. Soille, P. *Morphological Image Analysis-Principles and Applications*; Springer: Berlin/Heidelberg, Germany, 2003. [[CrossRef](#)]
47. Lavika, G.; Lavanya, B.; Panchal, P. *Hybridization of Biogeography-Based Optimization and Gravitational Search Algorithm for Efficient Face Recognition*; IGI Global: Hershey, PA, USA, 2019.
48. Becker, R.A.; Chambers, J.M.; Wilks, A.R. *The New S Language: A Programming Environment for Data Analysis and Graphics*; Wadsworth and Brooks/Cole Advanced Books & Software: Pacific Grove, CA, USA, 1988.
49. Bressemer, K.K.; Adams, L.C.; Erxleben, C.; Hamm, B.; Niehues, S.M.; Vahldiek, J.L. Comparing different deep learning architectures for classification of chest radiographs. *Sci. Rep.* **2020**, *10*, 13590. [[CrossRef](#)] [[PubMed](#)]