Article

Multi-Input Convolutional Neural Networks for Automatic Pollen Classification

Mihai Boldeanu ^{1,*} , Horia Cucu ¹ , Corneliu Burileanu ¹ and Luminița Mărmureanu ²
¹ CAMPUS Research Institute, University Politehnica of Bucharest, 060042 Bucharest, Romania; horiacucu@gmail.com (H.C.); corneliu.burileanu@upb.ro (C.B.)

² National Institute for Research and Development for Optoelectronics INOE 2000, 077125 Măgurele, Romania; mluminita@inoe.ro

* Correspondence: mishuboldeanu@gmail.com; Tel.: +40-737-467-738

Abstract: Pollen allergies are a cause of much suffering for an increasing number of individuals. Current pollen monitoring techniques are lacking due to their reliance on manual counting and classification of pollen by human technicians. In this study, we present a neural network architecture capable of distinguishing pollen species using data from an automated particle measurement device. This work presents an improvement over the current state of the art in the task of automated pollen classification, using fluorescence spectrum data of aerosol particles. We obtained a relative reduction in the error rate of over 48%, from 27% to 14%, for one of the datasets, with similar improvements for the other analyzed datasets. We also use a novel approach for doing hyperparameter tuning for multiple input networks.

Keywords: automatic pollen classification; convolutional neural networks; multi-input neural network; Rapid-E

Citation: Boldeanu, M.; Cucu, H.; Burileanu, C.; Mărmureanu, L. Multi-Input Convolutional Neural Networks for Automatic Pollen Classification. *Appl. Sci.* **2021**, *11*, 11707. <https://doi.org/10.3390/app112411707>

Academic Editors: Norbert Herencsar, Francesco Benedetto and Jorge Crichigno

Received: 15 November 2021

Accepted: 7 December 2021

Published: 9 December 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

Pollen is a powdery substance produced by plants during their reproductive season. While pollen is essential for plant survival and reproduction, certain species, especially those that use wind for pollen dispersal, are the cause of most seasonal allergies in humans [1].

The first reports of pollen as a possible cause for several illnesses date back to the beginning of the XIX century with researchers such as Bostock and Blackley [2]. Blackley experimented with several instruments to sample the air for pollen. He exposed glass slides coated in a glycerin mixture for a period of 24 h after which he analyzed them using a microscope. This was performed daily from April to August to record the number of pollen grains captured. These first experiments are the basis for what the field of palynology has researched for the following 150 years.

A major improvement to the study of air-borne pollen was the development of the Hirst volumetric trap [3]. His new system involved an air pump and a roll of tape coated in Vaseline which moved at a constant speed. This small change to the sampling method increased the amount of information that could be extracted by increasing the temporal resolution from daily to hourly concentration values that could be correlated with variations in weather. Although the instrument designed by Hirst is over 70 years old, it is still the standard when it comes to pollen monitoring.

The problem with the classical way of pollen monitoring has become apparent in recent years with the introduction of pollen monitoring networks. The increase in the number of pollen traps at a national or international level has forced aerobiologists to spend more and more of their time on pollen counting. Other important aspects of pollen monitoring are the lack of reliability of humans when it comes to pollen counting and classification [4] and the lag introduced by the need to have humans in the loop.

The lack of scalability of the Hirst-type systems is the reason why in recent years a move towards automated systems has started. The Rapid-E, an instrument developed by Plair (<http://www.plair.ch/Rapid-E.html>, accessed on 1 November 2021), is one such automated system that is capable of counting and analyzing particles autonomously. This device uses multiple lasers to measure the size, morphology and fluorescence spectrum of the aerosol particles.

Automatic pollen classification using machine learning techniques has been proposed as far back as 1968 [5]. The first and the most numerous attempts at pollen classification have been using microscope photography as the input for machine learning models.

While, so far, most of the work has been focused on microscope photography, many approaches have been explored throughout the years, such as fluorescence microscopy, flow cytometry, photoacoustic microscopy, fluorescence spectroscopy, Raman microscopy, Fourier-transform infra-red (FTIR), laser ablation mass spectrometry (LAMs) and elastically scattered light [6].

This work focuses on pollen classification using data from an automatic particle analyzer that uses air flow cytometry and fluorescence spectrum to characterize particles. The first use of deep learning techniques on such datasets was presented by [7], but were performed on a prototype device that was not widely available to use as a data source for automatic pollen monitoring. The first paper to introduce datasets from a commercially available device, Rapid-E, is [8], published in 2019. This area of research was expanded in [9] with better architecture selection and in [10] with the introduction of a new pollen dataset. This area of research is still in its early stages with many unknowns regarding the best use of the features provided by instruments such as the Rapid-E.

In this paper, we build upon previous work, first published in 2021 for the 44th International Conference on Telecommunications and Signal Processing (TSP) [9], by extending the study to an additional dataset that we recently collected and published [10] and by using a methodology for efficient hyperparameter search on large multiple input neural network architectures. On top of the contributions presented in [9], the main contributions of this paper are that we obtained a better performance compared to the previous state of the art [8] and our previous work [10] on the four datasets available from Rapid-E devices; this was achieved by using Bayesian optimization to perform a hyperparameter search to improve the architecture performance, and by conducting an in-depth analysis of the generalization capabilities of the models from one dataset to the others.

This paper is structured as follows: Section 2 describes the datasets, the types of architectures proposed for dealing with the heterogeneity of the data and the setup used for doing a hyperparameter search to optimize the architecture. Section 3 explores the results compared to the previous state of the art, proposes techniques that help with generalization of the trained models and discussions are made over the limitations of such systems and possible steps to resolve them. Finally, conclusions are drawn in Section 4.

2. Materials and Methods

In this section, the methodologies required to collect pollen measurements using a Rapid-E device are presented along with the algorithms used for developing classification models on this sort of data sources.

2.1. Datasets

This sub-section provides a description of the publicly available pollen datasets and the experimental setup used in creating them. All of the datasets were created in a similar fashion using the Rapid-E particle analyzer and are representative of four countries across Europe, Serbia, Lithuania, Switzerland and Romania.

2.1.1. Data Acquisition

The Rapid-E uses multiple lasers to obtain different characterizations of particles while they are in-flight, meaning that they are not deposited on slides or other capture media.

This enables the device to work continuously in remote areas with little to no interaction required from a human operator. On the other hand, this makes creating a database of pollen quite difficult, because the features extracted by the device are very difficult for the human expert to pin to a certain species after the measurement is taken. To overcome this challenge, datasets of pollen are created in the lab by exposing the device to pollen from one species at a time. The device provides three types of features for a particle using two lasers at different wavelengths of light. Using an infrared laser, the device is able to create a scattering image that contains information about the size and the texture of the analyzed particle. After the particle is detected with the infrared sensor, a UV laser is used to excite the particle to get it to produce the fluorescence spectrum. Because of the optical nature of the setup, problems can arise from misalignment. Particles can be missed if they are not well centered on the laser beam or particles may appear bigger than they actually are if two, or more particles stick together and behave as a single object. In the following section, a more in-depth look at the data is presented.

2.1.2. Dataset Statistics

Table 1 is an overview of the publicly available datasets. They are datasets that first appeared in [8,10] and were created with pollen from four different countries, Serbia (SAU-SRB) [8], Switzerland (SAU-CH) [8], Lithuania (SAU-LI) [8] and Romania (MARS) [10]. In the case of the MARS dataset, the measurements were made at Magurele center for Atmosphere and Radiation Studies (MARS) (44.34N, 26.01E), south of Bucharest, Romania, by some of the authors of this paper. They have different numbers of classes, or genera of pollen, and vastly different numbers of samples per class. This class imbalance has to be accounted for during training and evaluating the results of a classification model.

Table 1. Datasets overview.

Dataset	Number of Classes	Number of Samples	Minimum Number of Samples per Class
SAU-SRB ^a	14	85 k	985
SAU-LI ^a	11	399 k	16,114
SAU-CH ^a	10	50 k	1075
MARS ^b	13	105 k	3020

^a Dataset from [8]; ^b dataset from [10].

The datasets are varied in the number of classes and also the minimum, maximum and average number of samples per class. The pollen species present in each dataset also differ and are representative for the flora of each specific country.

Table 2 shows the pollen species present in each of the datasets. The species present in the datasets differ based on the biodiversity of the respective region. The presence of different species of pollen in the datasets makes a comparison among the results across all of them difficult, because some species of plants have very similar pollen in size, shape and fluorescence response. Therefore, for each of the datasets, the importance of each of the feature type differs.

Table 2. Datasets overview.

Plant Genus	# Samples SAU-SRB	# Samples SAU-LI	# Samples SAU-CH	# Samples MARS
Alnus	3944	47,816	6314	8148
Corylus	16,299	34,065	11,100	9073
Betula	12,784	64,461	14,858	–
Fraxinus	5596	–	4747	4310
Quercus	1889	34,019	–	5978
Populus	6077	22,273	–	16,923
Taxus	6006	–	4061	11,958
Pinus	–	31,144	3593	4068
Acer	985	16,114	–	–
Artemisia	15,208	39,546	–	–
Dactylis	1874	–	–	6902
Salix	10,136	36,635	–	–
Brussonetta	–	–	–	12,385
Carpinus	–	–	2377	–
Cedrus	1215	–	–	–
Cupressus	–	–	1387	–
Cynodon	1006	–	–	–
Fagus	–	–	1075	–
Festuca	–	27,556	–	–
Juglans	–	–	–	7038
Juniperus	–	45,888	–	–
Lolium Perene	–	–	–	3020
Morus	–	–	–	7674
Picea	2743	–	–	–
Platanus	–	–	–	8005
Poaceae	–	–	1110	–

2.1.3. Dataset Analysis

When developing neural network models that take advantage of very different data inputs, extra care has to be taken to make sure that such a model actually uses all the data inputs and does not over fit on some of them. This is why it is necessary to perform in-depth exploratory data analyses.

Figure 1 is a visualization of the features used for the pollen classification task. The three types of features are obtained from a single aerosol particle.

Figure 1. The three features provided by the Rapid-E instrument. From left to right: scattering image, fluorescence spectrum and lifetime signal.

The scattering image is a 120×24 tensor that shows the amount of laser light scattered by the particle at 24 different angles. The scattering images are obtained in a manner similar to a photocopy; the only difference is that the scanned object moves and not the light source and detector.

The fluorescence spectrum is a 32×8 tensor representing the intensity of the fluorescence response at 32 wavelengths at 8 moments in time measured at half a microsecond apart from each other.

Lifetime is a 64×4 tensor which corresponds to the time from when a UV laser pulse is shot until the fluorescence response of the particle. The lifetime signal is measured at 4 wavelength bins, 350–400 nm, 420–460 nm, 511–572 nm and 672–800 nm, and at a temporal resolution of nanoseconds. The physical information carried by the fluorescence lifetime signal refers to the average time the organic molecules in the pollen particle stay in their excited states before emitting a photon.

2.1.4. Data Pre-Processing and Filtering

Because of the limitations of the measurement device, such as optical misalignment and simultaneous multiple particle measurements, data pre-processing is required before training. One factor that is useful in filtering is the presence of a strong fluorescence signal for a particle. This is an important criterion by which filter, because all pollen particles are fluorescent to a certain degree. If the fluorescence feature is missing, it means that either we measured a particle that is not pollen (e.g., dust), or the particle was misaligned with the optical detectors and we have an invalid observation. These types of samples are removed from the dataset. Another feature that can be used for data filtering is the particle size, as this information can be extracted from the scattering image. The length of the raw scattering image tensor is given by the size of the particle and can be used to select only particles that are in the usual range for pollen (i.e., 10–90 micrometers).

2.2. Network Architecture

The architecture selected for the task of pollen classification using the features provided by the Rapid-E device is a multiple input convolutional neural network (CNN) [11], as depicted in Figure 2. Because the features are very heterogeneous, care has to be taken so that a neural network is able to use all the feature types. For this reason, a convolutional neural network [12] architecture was selected. In Figure 2, the architecture overview is presented. This architecture can be broken down into the following 4 main components: three feature extractors (FE), one for each of the three types of features, and a fully connected dense neural network [13] that uses the concatenated outputs of the feature extractors to make a classification.

The feature extractors can be further broken down into convolutional blocks. Each block is built using one or more identical convolution layers with ReLU activations [14],

followed by a batch normalization layer [15] and a max pooling layer [16]. After each max pooling layer the number of filters in the convolution layer is doubled to allow the model to cope with the reduction in the dimensionality introduced by the max pooling operation.

Figure 2. The convolutional network architecture used for pollen classification.

2.2.1. Feature Extractors

For this architecture, each of the feature extractors are very similar, with the major difference between them being that, for scattering (Figure 3) and spectrum (Figure 4), we used 2D convolutions, while, for the lifetime signal, we used 1D convolution layers, shown in Figure 5. The kernel size for the convolutional filters was 3×3 for the 2D filters and 3 for the 1D filters. The number of filters used was 32 at the first layer and doubled after each max pooling layer (i.e., 32, 64 and 128 filters). After each of the feature extractors, a flatten operation was applied to prepare the feature vectors for concatenation.

The feature extractors used in [9] have three convolution blocks, with each block composed of two convolution layers followed by a batch normalization layer and a max pooling layer. After each max pooling layer, the number of convolution filters doubles. After the last convolution block, we used dense layers that further helped to reduce the high dimensionality of the features. The presence of the fully-connected layers enabled the use of each feature extractor as a classifier on its own. This is a very important aspect of the architecture, because it allowed the feature extractors to be trained on their own. The importance of training the feature extractors independently is that this allows different learning rates, batch sizes and even optimizers for each feature type. The feature extractors can be “frozen” after training and a final training of the common trunk can be performed.

Figure 3. The scattering image feature extractor architecture, with C being the number of classes in each dataset.

Figure 4. The fluorescence spectrum feature extractor architecture, with C being the number of classes in each dataset.

Figure 5. The lifetime signal feature extractor architecture, with C being the number of classes in each dataset.

This architecture, with multiple inputs and outputs, allows pollen to be classified in two ways. A classification can be made by using a single feature type or all of the feature types. In order to evaluate the importance of each of the features in the classification task, intermediate output layers were added after each of the feature extractors. These outputs use the latent representation of the features to make a classification using just one of the features. This was necessary, because each of the datasets contained different classes of pollen that have different degrees of similarity between classes for each of the feature types. The advantage in having intermediate output layers is that each of the feature extractors can be trained independently should they require a different learning rate, batch size, or number of training epochs.

2.2.2. Common Trunk

After the feature extractors, a concatenation layer was used to combine the information from the hidden layers. From this latent representation, dense layers were used to obtain a classification of the combined information tensor. The network trunk was composed of dense layers with drop-out [17] and ReLU activation functions. The dense or fully connected part of the network was composed of 3 layers that had 256 and 128 neurons for the first and second layer, respectively, and the last layer had a different number of neurons depending on the number of classes in each dataset. The drop-out layers help

with regularization and prevent over-fitting of the training data. The model drop-out was activated only while training; the drop-out layer was deactivated at evaluation so that the entire network was used. The loss selected for the classification task was categorical cross-entropy [18]. The overall loss was computed as the weighted sum of the losses of each of the feature extractors and the main output loss. This forced the model to use the information from all the feature types and not try to over-fit on one just one type of feature. This is important because some of the pollen classes are very similar for some of the feature types and if only one loss had been used the network would have ignored those features.

2.2.3. Training

While the task of training a neural network is straightforward in most cases, when dealing with a modular, multiple-input design, certain issues arise regarding the best learning strategy. There are three options available when selecting a learning strategy: (i) the entire model is trained at the same time; (ii) we first train the feature extractors, then we only train the common dense trunk, keeping the FE weights “frozen”; or (iii) we train the feature extractors, then we train entire model, with pre-initialized FEs.

The advantage of training the full model is that an optimal weighting of the importance of each feature type is performed. The model learns what features are most important for a good classification and relies on them more. While this can make for a better model in theory, in practice, we saw that training is more difficult because of the higher number of layers that have to be updated at each back-propagation step. The models tend to train slowly because of the large number of layers; further, the feature extractors that train faster tend to over-fit while other ones are not fully trained yet. This makes selecting the learning rate, batch size and number of epochs for training more challenging.

In contrast, when training each feature extractor independently, most of these issues disappear. For each of the feature extractors, training can be conducted for a different number of epochs and with a different learning rate to obtain an optimal solution for that feature type. This gives better feature extractors that help in getting a better overall classifier. Another advantage is that, when trying to improve the model by selecting good hyperparameters, the search space is reduced by looking at each feature extractor independently as a stand-alone model.

2.2.4. Hyper-Parameter Tuning

A standard procedure to increase the performance of any machine learning algorithm is selecting good hyperparameters for the specific task. The number of hyperparameters that are available for tuning depends on the complexity of the proposed approach, with convolutional neural networks having a very large number of possible values to choose from. Finding a good hyperparameter tuple in a very large search space is difficult and computationally expensive; therefore, the task was split into multiple smaller problems. Thanks to the modularity of the network and the possibility of training each component independently, the hyperparameter search was conducted separately for each feature extractor. This reduced the dimensionality of the search space by orders of magnitude and increased the chances that good hyperparameters could be found.

In Table 3, the values selected for the hyperparameter search are presented. For most parameters, the values were selected considering a linear range, except for the learning rate, which used a logarithmic scale to better probe the search space. The last column of the table presents the setup used in [9].

Table 3. Hyperparameters.

Hyperparameter	Possible Range	Used in [9]
learning rate	0.01–0.000001	0.001
batch size	16–256	64
size dense layer	32–1024	256
drop-out rate for dense layer	0–0.9	0.25
filter size	1–11	3
# of conv. filters (first layer)	4–128	32
# of conv. layers/cells	1–7	2
# of cells	1–4	3

symbol is used in place of number.

When searching for a good combination of hyperparameters, a grid search [19] is guaranteed to eventually give the best results, if a small-enough step is selected. The only problem with this approach is that it takes huge amounts of time to train all the different variations of the models. This approach is only applicable if the number of hyperparameters is really low.

Another approach is using a random search. This method was shown to be good enough with a much smaller impact on training time [20]. Hyperparameter values are chosen at random from the available ranges and a model is trained. After training, the model is evaluated and the parameter and score are saved.

Even better than a random search is a guided approach, such as Bayesian optimization [21]. This approach tries to optimize over the search space by first constructing a posterior distribution of functions that best describes the function to be optimized. Then, as the number of observations grows, the posterior distribution improves and the algorithm becomes more certain of which regions in the parameter space are worth exploring and which are not. The Bayesian optimization was performed using a Python package developed by [22]. This package was used to first randomly sample the search space to initialize the algorithm. After this first step, the package recommended the hyperparameters based on metrics that balanced exploration versus exploitation of the known features of the search space.

Hyperparameter tuning was performed independently for each feature extractor, with each tuning run consisting of 10 initial random selections followed by 200 guided selections. As can be seen in Table 4, the parameters selected after the search are very different depending on the feature in question. For the scattering image feature extractor, a higher learning rate was selected than for the other two. The selected batch size was more than double for the spectrum feature than for the lifetime one. The number of neurons in the dense part of the feature extractors appears to be correlated with the drop-out rate. The filter kernel size chosen was 5×5 for scattering and spectrum and 3 for the lifetime signal. Looking at the number of filters per layer, we can see that, even though the possible range was 4–128, only the lifetime feature extractor went in the direction of many filters, with the other two using less filters than the initial architecture.

Table 4. Best hyperparameters. Hyperparameter tuning was performed independently for each FE; thus, the best configuration of hyperparameters is shown independently for each one.

Hyperparameter	Scattering	Spectrum	Lifetime
learning rate	0.001	0.0001	0.0001
batch size	175	248	112
size dense layer	362	718	614
drop-out rate dense layer	0.3	0.8	0.6
filter size	5	5	3
# of conv. filters (first layer)	28	14	108
# of conv. layers/cells	3	2	1
# of cells	2	1	3

symbol is used in place of number.

3. Results and Discussions

The Results and Discussions section is structured as follows: The first two sub-sections focus on the results of the presented model before and after the hyperparameter search. Section 3 examines the sources of error in classification, while the last part analyzes the generalization capabilities of the models from one dataset to another.

3.1. Classification Results

In this section, we evaluate the results of the trained model and compare to the previous work performed on this type of classification for all four datasets. In Table 5, the results of our trained network are presented in comparison to the results from [8].

Table 5. Model performance (accuracy).

Dataset	SAU-SRB	SAU-Li	SAU-CH	MARS
Baseline ^a	74%	73%	80%	-
Initial architecture ^b	77%	84%	85%	76%
Scattering image only	53%	60%	57%	59%
Fluorescence spectrum only	49%	70%	45%	50%
Lifetime signal only	60%	57%	67%	39%

^a Results from [8]; ^b results from [9].

The results presented in Table 5 show an improvement over the baseline [8] on all datasets, with the highest increase in accuracy for the SAU-LI dataset. The results analysis can be split into two parts based on the usage of features. SAU-SRB and SAU-CH used all three feature types and can be compared directly with the results from this work. As presented in Table 5, our approach improved over the current best. In respect to SAU-SRB, we reduced the error rate from 26% to 23% and, compared to SAU-CH, from 20% to 15%. The greatest improvement, obtained in respect to the SAU-CH dataset, is in part due to the lower number of classes in this dataset.

When comparing with the results on the SAU-LI dataset, extra care has to be taken into account as the original paper only used two of the three feature types (i.e., spectrum and scattering). In our approach, all three features were used, leading to the best improvement compared to the other datasets. An improvement is still present even if we use just the two features, as in the original work. This was performed by multiplying with zeros the lifetime feature and doing a separate training and testing run. In this comparison, we obtained an accuracy of 78% by using just the two features. The takeaway message is that having

more features improves classification even though these extra features appear noisy to the human eye.

In Table 5, rows 3–5 show the results obtained by performing the classification with one feature type at a time. This was possible thanks to the added intermediary outputs after each feature extractor. The results show that the most important feature depends on the dataset and on the relationship among the pollen classes in the dataset. For example, if we have many classes of pollen that have similar size and shape, the scattering image is less useful for classification (see dataset SAU-SRB, for example). Another important thing is that the features appear to be complementary, the combined results of the classification being well above the performance of any individual feature extractor.

3.2. Classification after Hyper-Parameter Tuning

In this section, we evaluate the results of the best model obtained in the hyperparameter search and compare it with the results from previous works on all four datasets.

In Table 6, we can see that all models gained accuracy using the best hyperparameter configuration found. The gains were not equal across the board with most of the improvement having been obtained by the models trained on SAU-SRB and MARS. These were the most difficult datasets when looking at the number of classes. The increase in performance is more visible when looking at each individual feature extractor. Comparing rows 5–7 of Table 6 with rows 4–6 from Table 5, we can see that all the feature extractors were much better at making classifications. Examining the error rate for the scattering image feature across all four datasets, we obtained a drop from an average of 42% in the initial case to 38% using the best architecture found. For the fluorescence spectrum, the improvement was even greater, from an average error rate value of 46% to 37%. Finally, for the lifetime signal, the average error rate across the datasets decreased from 44% to 40%. In an ideal scenario, these improvements should translate to a similar improvement in the combined model. However, what we observed in practice is that the combined model error rate decreased but at diminishing returns. With the fine tuned architecture, compared to the initially proposed architecture, we obtained a relative reduction in the error rate of 13%, from 23% to 20%, for SAU-SRB; of 12%, from 16% to 14%, for SAU-LI; of 13%, from 15% to 13%, for SAU-CH; and of 20%, from 24% to 19%, for MARS.

Table 6. Model performance after hyperparameter tuning (accuracy).

Data-Set	SAU-SRB	SAU-LI	SAU-CH	MARS
Baseline ^a	74%	73%	80%	-
Initial architecture ^b	77%	84%	85%	76%
Best architecture	80%	86%	87%	81%
Scattering image only	58%	62%	61%	64%
Fluorescence spectrum only	56%	72%	66%	58%
Lifetime signal only	68%	59%	72%	41%

^a Results from [8]; ^b results from [9].

The diminishing returns observed in the model accuracy are explained by the fact that each feature extractor worked well for a sub-set of the whole dataset and the combined model worked well on the union of these sub-sets, that were well classified by each feature extractor. When the accuracy of a feature extractor rises this means that the sub-set of samples that are correctly classified increases. When examining all three feature extractors globally, we did not obtain the same increase in performance, because there was an increase in the overlap of samples that were classified by each feature extractor, as can be seen in Figure 6.

Figure 6. Performance of the combined model before (a) and after (b) hyperparameter search. The black set represents the entire dataset and each of the three smaller sets (red, yellow and green) is a sub-set of the samples that were classified correctly by using each feature type. The green area is the sub-set of the samples classified correctly by the combined model.

From the point of view of the combined model, the improvement seen by each feature extractor translates to a higher confidence, rather than a higher overall accuracy.

Considering at the results of the hyperparameter experiments, we can see that part of the error in classification is inherent to each of the datasets and difficult to overcome without an in depth analysis of the source of such errors. In the following subsection, the classification error is described using the confusion matrix.

3.3. Analysis of Error Sources

While the results at a dataset level are important on their own, without an attentive analysis of the distribution of errors across the dataset, some information would be missing. This is where the confusion matrix comes in to give a more accurate idea of what classes are the main sources of errors.

When examining the confusion matrix (Figure 7) for one of the datasets, SAU-LI, a clear pattern emerges. We can see a higher confusion for some pollen types, with the others having the error more uniformly spread out. We can also see that most of the errors in the classification regarded just four classes: *Alnus*, *Corylus*, *Betula* and *Quercus*.

Figure 7. Confusion matrix created on the test set from the SAU-LI dataset. Values are percentages.

Figure 8 (left) presents microscope slide pictures of the pollen classes that have the highest confusion (a–d), revealing the similarity in size and shape, in comparison with two other pollen species (e,f). All the highly confusable genera have distinctive “bumps” on the surface and are very similar in size. This would translate to obtaining a similar scattering image among these species. When analyzing the fluorescence spectra for the same four classes (Figure 8, right), we can see that they are also very similar. This makes distinguishing among these classes exceedingly difficult compared to other species, such as *Artemisia* and *Pinus*, that are quite distinct in size, shape and fluorescence spectrum and are quite easy for the model to classify.

Figure 8. On the left, images of different pollen species on microscope slides [23]: (a) *Alnus glutinosa*, (b) *Betula pendula*, (c) *Corylus avellana*, (d) *Quercus robur*, (e) *Ambrosia artemisiifolia*, (f) *Pinus sylvestris*, (g) *Juglans regia*, (h) *Acer* sp., (i) *Poa*., (j) *Taxus baccata*, (k) *Ulmus* sp. and (l) *Fagus sylvatica*. On the right, the average fluorescence spectrum for some of the classes present in SAU-LI.

In Figure 9, the confusion matrix was built from the classifications made with a single feature at a time. The patterns observed in Figure 7, for the combined model, were present in the results of all the feature extractors' classifications.

Figure 9. Confusion matrix created on the test set from the SAU-LI dataset with feature extractor classification; from left to right: scattering, spectrum and lifetime. Values are percentages.

3.4. Inter Dataset Comparison

All the experiments presented above were conducted with models trained and tested on data from the same dataset; both the train and test sub-sets were generated on the same device. This was conducted because each dataset has particular classes that are not common

across all four data sources. The problem with training on a dataset created on just one device is that it increases the risk of over-fitting. In this section, we investigate the classification results obtained for models trained on one dataset's training set and tested on another dataset's test set. The expected error was higher because some classes are not present in both sets but the common pollen should ideally be well identified. The following experiments look at the generalization capabilities of the models.

Figure 10 shows a clear case of over-fitting causing a mode collapse, with all the predictions made by the trained models falling in just few classes. While this was expected for the classes not common to both datasets, this also happened to the common ones. The cause of the over-fitting is the fact that each dataset was constructed on just one device. The models learned the instrument specific noise and, when presented with samples from a different device, they were unable to make an accurate classification.

Figure 10. Left: Confusion matrix created on the test set from the SAU-CH dataset with combined model trained on the training set from SAU-LI. Right: Confusion matrix created on the test set from the SAU-LI dataset with model trained on the training set from SAU-CH. Values are percentages.

Figure 11 shows a more detailed examination at the same scenario but expanded to each of the feature extractor sub-models. While, for most feature extractors, the mode collapse was around two or three classes, in the case of the spectrum feature for the SAU-CH dataset, the mode collapse was a single class, indicating that the spectrum feature suffered the most from one device to the other.

Figure 11. Cont.

Figure 11. Left: Confusion matrix created on the test set from the SAU-CH dataset with feature extractor models trained on the training set from SAU-LI. From top to bottom: scattering FE, spectrum FE, lifetime FE. Right: Confusion matrix created on the test set from the SAU-LI dataset with feature extractor models trained on the training set from SAU-CH. From top to bottom: scattering FE, spectrum FE, lifetime FE. Values are percentages.

4. Conclusions

In this paper, an automatic pollen classification model is proposed. It improves over the current state of the art. Although the improvement was significant for all of the datasets, there are still questions to be answered regarding the best use of the features provided by instruments such as the Rapid-E device. After an extensive hyperparameter search, we obtained a relative reduction in the error rate of 13%, from 23% to 20%, for SAU-SRB; of 12%, from 16% to 14%, for SAU-LI; of 13%, from 15% to 13%, for SAU-CH; and of 20%, from 24% to 19%, for MARS, over our initial results presented in [9]. Moreover, we obtained an overall relative reduction in the error rate of 23%, from 26% to 20%, for SAU-SRB; of 48%, from 27% to 14%, for SAU-LI; of 35%, from 20% to 13%, for SAU-CH; and over the previous state of the art, from [8]. These results show the importance of finding good hyperparameters for each specific task and that using a guided method for tuning the models can be more efficient.

We investigated the source of the model error and found that, for certain classes, the pollen is very difficult to differentiate using the available features. The classes that cause the most error are *Alnus*, *Betula*, *Corylus* and *Quercus*, because the shape and size of the particles and the florescence spectrum are very similar. This makes it difficult for a model to be able to make a good classification.

Our final contribution is related to the analysis of the inter-dataset error. This sort of error is very important to study, because it measures the generalization capability of a model when presented with new data from different devices. We found that the models tend to over-fit the training data. This issue can be explained by the fact that each of

the datasets was created on different Rapid-E devices, with just one device used for each dataset. The models learned the instrument noise at the same time with useful information used in classification and, when presented with data from a different instrument, suffered a total mode collapse.

The next steps in finding better models for pollen classifications using the information from Rapid-E devices are as follows: (i) creating more datasets that have data from as many devices as possible with more overlap in the genera of plants present in each dataset; (ii) performing an advanced network architecture search to find novel model designs that would make better use of the temporal component of the scattering image, fluorescence spectrum and lifetime; (iii) designing new training strategies that are able to take advantage of data from multiple datasets with different classes; and (iv) for actual adoption in the pollen community, these models would have to be validated over full pollen seasons and cross-validated with manual pollen count obtained from Hirst-type volumetric traps. Further research is required for a better understanding of the source of most errors in the model.

Author Contributions: Conceptualization, M.B. and L.M.; methodology, M.B. and H.C.; software, M.B.; validation, M.B. and L.M.; resources, L.M.; data curation, L.M.; writing—original draft preparation, M.B.; writing—review and editing, H.C., C.B. and L.M.; visualization, M.B. All authors have read and agreed to the published version of the manuscript. The contributions of all authors are considered to be equal.

Funding: This research study received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available on request from the corresponding author, M.B.

Acknowledgments: This work was supported by the Romanian National Core Program Contract No. 18N/2019. Further funding was provided by the European Regional Development Fund through the Competitiveness Operational Programme 2014–2020 POC–A.1–A.1.1.1-F-2015, project Research Centre for Environment and Earth Observation CEO-Terra, 328SMIS code 108109, contract No. 152/2016 and by a grant of the Romanian Ministry of Education and Research, CCCDI—UEFISCDI, project number PN-III-P2-2.1-PED-2019-2278, within PNCDI III.

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

Abbreviations

The following abbreviations are used in this manuscript:

CNN	convolutional neural network
FE	feature extractor
MARS	Romanian dataset first presented in [10]
SAU-CH	Swiss dataset first presented in [8]
SAU-LI	Lithuanian dataset first presented in [8]
SAU-SRB	Serbian dataset first presented in [8]

References

1. Xie, Z.J. Advances in the clinical and mechanism research of pollen induced seasonal allergic asthma. *Am. J. Clin. Exp. Immunol.* **2019**, *8*, 1–8.
2. Comtois, P. The experimental research of Charles H. Blackley. *Aerobiologia* **1995**, *11*, 63–68. [[CrossRef](#)]
3. Hirst, J.M. An automatic volumetric spore trap. *Ann. Appl. Biol.* **1952**, *39*, 257–265. [[CrossRef](#)]
4. Gottardini, E. Sampling bias and sampling errors in pollen counting in aerobiological monitoring in Italy. *J. Environ. Monit.* **2009**, *11*, 751–755. [[CrossRef](#)]
5. Flenley, J.R. The problem of pollen recognition. In *Problems of Picture Interpretation*; Clowes, M.B., Penny, J.P., Eds.; CSIRO: Canberra, Australia, 1968; pp. 141–145.
6. Holt, K.A. Principles and methods for automated palynology. *New Phytol.* **2014**, *203*, 735–742. [[CrossRef](#)]

7. Crouzy, B. All-optical automatic pollen identification: Towards an operational system. *Atmos. Environ.* **2016**, *140*, 202–212. [[CrossRef](#)]
8. Šaulienė, I. Automatic pollen recognition with the Rapid-E particle counter: The first-level procedure, experience and next steps. *Atmos. Meas. Tech.* **2019**, *12*, 3435–3452. [[CrossRef](#)]
9. Boldeanu, M.; Cucu, H.; Burileanu, C.; Mărmureanu, L. Automatic Pollen Classification Using Convolutional Neural Networks. In Proceedings of the 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 26–28 July 2021; pp. 130–133.
10. Boldeanu, M.; Marin, C.; Ene, D.; Mărmureanu, L.; Cucu, H.; Burileanu, C.L. MARS: The First Romanian Pollen Dataset using a Rapid-E Particle Analyzer. In Proceedings of the 2021 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), Bucharest, Romania, 13–15 October 2021; pp. 145–150.
11. Ahmed Raza, S.E. MIMO-Net: A multi-input multi-output convolutional neural network for cell segmentation in fluorescence microscopy images. In Proceedings of the 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017), Melbourne, VIC, Australia, 18–21 April 2017; pp. 337–340.
12. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [[CrossRef](#)] [[PubMed](#)]
13. Pal, S.K. Multilayer perceptron, fuzzy sets, and classification. *IEEE Trans. Neural Netw.* **1992**, *3*, 683–697. [[CrossRef](#)]
14. Agarap, A.F. Deep learning using rectified linear units (relu). *arXiv* **2018**, arXiv:1803.08375.
15. Ioffe, S. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the ICML'15: 32nd International Conference on International Conference on Machine Learning, 2015, Lille, France, 6–11 July 2015; Volume 37, pp. 448–456.
16. Nagi, J. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In Proceedings of the 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Kuala Lumpur, Malaysia, 16–18 November 2011; pp. 342–347.
17. Srivastava, N. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
18. Zhang, Z. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, QC, Canada, 3–8 December 2018.
19. Shekar, B.H. Grid Search-Based Hyperparameter Tuning and Classification of Microarray Cancer Data. In Proceedings of the 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP), Sikkim, India, 25–28 February 2019; pp. 1–8.
20. Bergstra, J. Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.* **2012**, *13*, 1–8.
21. Wu, J. Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization. *J. Electron. Sci. Technol.* **2019**, *17*, 26–40.
22. Nogueira, F. Bayesian Optimization: Open Source Constrained Global Optimization Tool for Python. 2019. Available online: <https://www.github.com/fmfn/BayesianOptimization> (accessed on 1 October 2021).
23. Denisow, B. Pollen grains as airborne allergenic particles. *Acta Agrobot.* **2015**, *4*, 281–284. [[CrossRef](#)]