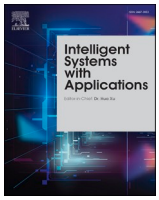
Intelligent Systems with Applications 19 (2023) 200264

Contents lists available at ScienceDirect

Intelligent Systems with Applications

journal homepage: www.journals.elsevier.com/intelligent-systems-with-applications



Multisource neural network feature map fusion: An efficient strategy to detect plant diseases

Guillaume Heller [a,b,*], Eric Perrin [a], Valeriu Vrabie [a], Cedric Dusart [b], Marie-Laure Panon [c], Marie Loyaux [c], Solen Le Roux [b]

[a] CRESTIC EA 3804, Université de Reims Champagne Ardenne, 51097 Reims, France
[b] Segula Technologies, 51100 Reims, France
[c] Comité Interprofessionnel du vin de Champagne, 5 Rue Henri Martin, CS 30135, 51204 Epernay, France

ARTICLE INFO

Keywords:
Feature aggregation
Ensemble model
Image classification
Multispectral imaging
Grapevine yellow

ABSTRACT

The use of RGB cameras or multispectral imaging systems can provide a wide range of applications for crop monitoring, plant phenotyping and disease detection. Although several approaches have been proposed, they increasingly use convolutional neural network-based architectures, which have, however, become increasingly cumbersome for improving classification results and difficult to train with few labeled data. Other increasingly popular approaches consist of using an ensemble of convolutional neural networks, in which each model solves a different problem. Since the inference is time- and resource-consuming due to the execution of multiple models, recent works have focused on transferring knowledge from an ensemble of models to a compact model to obtain better performance. In this paper, we propose an original approach that improves both accuracy and speed by reusing feature maps extracted by heterogeneous models from different data. Linked to each model, a transformation block allows keeping the correct number of feature maps and changing their dimension if necessary. To generate the feature maps, we only need the first layers of the ensemble models, thus taking advantage of ensemble learning methods, while adding only a few layers of a second model dedicated to aggregation of features. This approach allows an ensemble of models to be combined with different architectures that can process different data, such as several representations of the same input image or multispectral images, while being fast enough at the inference stage. This approach is adapted to hierarchical classification tasks by re-exploiting the same feature maps with different transformation blocks, offering accuracy gains in tasks not handled by the ensemble model. The results are provided for the PlantVillage dataset, with RGB images converted to three different color spaces, and for a custom Grapevine Yellow dataset, with multispectral images acquired with two different multispectral cameras.

1. Introduction

Agriculture is increasingly affected by global climate change. Among the effects observed in recent years in France and worldwide, new plant diseases are quickly developing and spreading, such as yellow beet virus, yellow rust, brown rust, Bois Noir, and Flavescence Dorée. Numerous studies have investigated the potential use of RGB imaging systems for plant disease diagnosis using deep learning techniques (Ahmad et al., 2023). The most common deep learning models are convolutional neural networks (CNNs) since they have achieved increasingly impressive scores, especially in classification tasks, in several real-world applications (Li et al., 2022; Zhang et al., 2023). These models have been suggested in various applications linked to precision agriculture (Coulibaly et al., 2022), including disease diagnosis and crop classification, as well as disease severity prediction and crop loss estimation (Kundu et al., 2022).

An increasing number of studies focus on the detection of plant disease using multispectral imaging, especially Flavescence Dorée, one of the two main Grapevine yellow diseases, has no cure and has rapidly spread over the years (Albetis de la Cruz, 2018; Al-Saddik et al., 2017). Although multispectral imaging generally allows improvement in the disease detection accuracy, there is another interest in its use. Multispectral imaging is important for the detection of Flavescence Dorée disease since we consider the disease on white grape varieties for which the symptoms are less visible than on red grape varieties. Moreover, the

* Corresponding author.
E-mail address: guillaume.heller@univ-reims.fr (G. Heller).

https://doi.org/10.1016/j.iswa.2023.200264
Received 4 June 2023; Received in revised form 19 July 2023; Accepted 1 August 2023
Available online 4 August 2023

Abbreviations

CNN	: Convolutional Neural Network
KD	: Knowledge Distillation
PCA	: Principal Component Analysis
UAV	: Unmanned Aerial Vehicle
GY	: Grapevine Yellow

first symptoms of this disease are not easily observable since they do not occur in the visible range. It may take several days between the first sign of stress that the plant undergoes and the manifestation of the disease on the leaves. Multispectral imaging can therefore be useful for early disease detection and to simplify their processing, and vegetation indices have been proposed as biophysical indicators (Albetis de la Cruz, 2018). However, to construct vegetation indices, multispectral images need to be realigned, as multispectral cameras are built as a matrix of sensors, each acquiring an image in a specific spectral band. Multispectral images must therefore be utilized as such (Al-Saddik et al., 2017), and to avoid aligning them, it will be interesting to move toward the fusion of features extracted by several CNN models, each dedicated to a specific band.

Despite multiple advantages of CNN models and recommendations provided for farmers and researchers to help develop appropriate tools for plant disease management (Ahmad et al., 2023), these architectures have become increasingly cumbersome to improve classification results, making their use impractical for applications requiring real-time processing or calculations on onboard systems with limited capacities. In addition, combining multiple CNNs will weigh down the architecture of the global model. Such cumbersome architectures can be hard to train with few labeled data and can be affected by the vanishing gradient problem. For example, a cumbersome ResNet architecture was employed and achieved promising results on RGB images (Boulent et al., 2020); however, the dataset was too small to conclude the robustness of this model. Such a model only represents a part of the global model that will have to work with multispectral images. This is another major challenge that needs to be addressed, as there is generally no large, annotated dataset for real applications, especially considering climate changes that impact symptom expression, as in the case of Flavescence Dorée.

To overcome this last problem, ensemble learning methods have been proposed by training reasonably sized models and using all their predictions to improve the final result (Yang et al., 2023). These approaches are based on the variability of features extracted by each model from the same data under similar conditions. Depending on the initial weights, the data augmentation options or the order of the batches, two models sharing the same architecture may not learn in the same way because during the training phase, we randomly apply data augmentation options. Therefore, similar architectures will be trained on slightly different images, which will cause differences in their parameters and thus generate slightly different feature maps. The use of multiple models usually provides better generalization capacities; however, it is time- and resource-consuming to fully train multiple deep neural models. Beyond the difficulties during the training stage, such ensemble approaches are difficult to use in the inference stage in real-world applications due to expensive computation time.

Unlike the research axis that consists of having increasingly precise and cumbersome models, other studies have focused on the design of models capable of operating in a short computing time while avoiding loss of accuracy. Among the existing models imagined for this purpose, networks belonging to the EfficientNet family (Tan & Le, 2020) are currently the most popular. By optimizing the depth, width and resolution at each layer, such networks achieve an interesting balance between speed and accuracy on popular datasets such as Cifar-100 (Krizhevsky, 2009) and ImageNet (Russakovsky et al., 2015). Other approaches, such as pruning, have focused on compressing high-performance cumbersome models to obtain a compact model with much fewer parameters (Blalock et al., 2020).

Hoping to build accurate, lightweight, and fast operating models, other studies have sought to transfer "knowledge" from a complex model to a compact model. To optimize the performances of the compact model, the main idea is to train it to provide the same predictions as the robust model rather than directly training it without the model to imitate (Ba & Caruana, 2014; Urban et al., 2017). These studies are aimed at "mimicking the function learned by ensemble selection", which is a previously introduced method (Caruana et al., 2004) for ensemble learning that was generalized in 2015 with the knowledge distillation (KD) approach. KD introduced a new way to transfer knowledge from an ensemble model to a compact model using soft labels (Hinton et al., 2015).

Our motivation is consistent with that of KD, one of the most widespread concepts for knowledge transfer between models currently. We propose to use multiple models capable of processing heterogeneous data, such as several representations of the same image or multispectral images, to enhance precision while having a model capable of quick inference. Instead of proposing a new way of transmitting knowledge during the training phase, we reuse the features extracted by the different models as input to a compact model directly during the inference phase without adding too many calculations. The objective is to reduce the redundancy present in the first layers of classical CNNs by taking advantage of a combination of varied and complementary feature maps through a new feature aggregation method. The main contributions of this work are presented as follows:

- We propose a feature aggregation method to exploit low-level knowledge acquired by an ensemble of models within a new compact model. The overall model performance is improved due to the use of a multitude of complementary heterogeneous features. The redundancy of features extracted by a single cumbersome model is replaced by the complementarity of heterogeneous features extracted by multiple models.
- At the inference stage, we only exploit the first layers of each model from the ensemble to extract the feature maps, which saves time and resources. We then combine the feature maps via a transformation block and use them as input for a compact model to make the classification. The overall model is comparable to an ensemble model in terms of accuracy but much faster to be viable during the inference phase.
- The proposed overall model is adapted to the use of high-dimensional data such as multispectral images. Each model from the ensemble focuses on the extraction of features from a specific band, whereas the compact model, which proceeds to the classification, uses the best features extracted from each band. In addition, it is not necessary to process the input images, for instance, to realign them, which also saves time. The last two experiments show that we can disregard the parallax effect between different imaging sensors, which is particularly interesting for real-world applications.
- We propose a transformation block specific to each model included in the ensemble, which allows the use of models with different architectures, unlike classical KD methods based on feature maps. It is thus possible to select the best models of the ensemble without any constraint on the architecture. For instance, we can combine compact models for simple image processing, such as RGB images, and complex, heterogeneous models for harder images, such as multispectral images.
- The proposed strategy can be applied to improve the accuracy on a subproblem of a main problem addressed by the ensemble models for a hierarchical classification task. For example, with the PlantVillage dataset, the main addressed problem could be the classification of different plants, whereas a subproblem could be the identification of

diseases for a specific plant. Our method allows us to be more efficient on the subproblem due to an ensemble of models designed to address the main problem. Moreover, since the main problem is simpler, the training is faster, which is interesting because there will be no other training steps for the models from the ensemble.

Subsequently, we will refer to the ensemble models that extract the feature maps as "level 1 models" and the compact model that receives these low-level feature maps and uses them for the inference as the "level 2 model". Our proposed global model is formed from these two-level models and the respective transformation blocks.

2. Related work

Among workable solutions, knowledge transfer is a relevant approach. One of the first studies carried out to efficiently transfer knowledge from an ensemble of models into a compact model is (Bucila et al., 2006). The authors, rather than directly training their compact, student model on a small training dataset, use an ensemble of models to annotate a large unlabeled dataset and then train a student model on that dataset. As the ensemble model labels the data, the student model will seek to produce similar predictions, thus achieving better performance than with direct training on the initial annotated dataset.

The popular KD approach was introduced in Hinton et al. (2015) to take advantage of the performance of ensemble models. The authors use the input of a softmax layer as soft labels instead of using hard labels. Thus, they can exploit additional information, assuming that the membership probabilities to the other classes granted by the teacher model is relevant information for understanding and mimicking the model's reasoning.

Further research has focused on the use of complementary labels to facilitate the transfer of knowledge. In Romero et al. (2015), the authors proposed using what they refer to as "hints", which correspond to the output of a hidden layer, to force the model to generate similar feature maps. This solution is designed to transfer knowledge from a shallow and wide model to a thin and deep model. The authors also show that it is more relevant to use inner layers as hints rather than classification targets.

In Furlanello et al. (2018), the authors iteratively retrained the same model by reusing previously generated features and obtained a gain in accuracy compared to the initial model.

More recently, a compact model composed of multiple branches as student models was proposed (Asif et al., 2020). The number of branches of the CompNet is equal to the number of models in the Teacher Ensemble Network. KD-loss based on Kullback–Leibler divergence and mean-squared error is utilized between each branch and the corresponding model in the teacher ensemble to guide the training. This loss is also applied one last time after all the features are grouped in both models.

Another research axis for transfer knowledge attempts to work at the feature-map level instead of the label level to transfer information that is more specific. The recent work in Heo et al. (2019), is aimed at distilling the knowledge within the activation boundaries, i.e., the hyperplanes that allow the separation among the classes. The authors propose a solution to transfer between spaces of different dimensions due to a connector function that typically consists of a fully connected layer or a 1×1 convolutional layer, possibly associated with a batch normalization layer. Introduced in Romero et al. (2015), the connector function transforms the response of a student layer into a vector of the same size as a teacher response vector. Due to this transformation, the authors can use an alternative loss and therefore transfer knowledge between different models. The authors achieve better results than with the initialization with ImageNet weights, which is a guarantee of quality in the transfer knowledge problem. This approach, however, is designed to transfer knowledge from a single teacher model.

In Zagoruyko and Komodakis (2017), the authors use an attention mechanism to transfer knowledge at different points of a teacher model as the information varies depending on the position. This solution, however, requires similar architectures so that the student model can use the information.

In Park and Kwak (2019), the authors propose transferring knowledge from an ensemble of similar models to another model with the same architecture. They use a nonlinear transformation layer and an autoencoder reconstruction loss between the feature maps of teacher and student models. An alternative method is also proposed with an iterative transfer instead of a parallel train from a set of models. In this sequential strategy, the former student model becomes the teacher for the next model. However, this solution also requires the same architecture for each model.

In Ji et al. (2021), automatic attention-based feature matching is proposed. The key idea is to force the features extracted by the student model to match those extracted by the teacher model with whom they share the greatest similarity. Rather than using predefined links, the authors propose to automatically determine the best matches according to a query-key concept previously introduced in Xu et al. (2016). Different experiments are run, notably for networks with different architectures. This concept is, however, suitable for knowledge transfer from a single model.

In recent years, feature aggregation methods have improved model performances for computer vision tasks. The most common example is probably the use of multiscale features, introduced in Lazebnik et al. (2006). Multiscale features are primarily obtained by using multiscale image inputs and extracting multiple features as discussed in Gong et al. (2014). There are other fusion strategies, such as multimodal information (with data of different types), multifocus fusion (the same data with different focal lengths), multitemporal fusion or multiview fusion, as presented in (Zhang et al., 2020) for neuroimaging. Fusion strategies are widespread in the processing of medical images, as they are subject to similar difficulties in smart agriculture applications, given the reduced training datasets or visually close classes (Wang et al., 2021).

Recent state-of-the-art concepts have also been applied to feature aggregation methods. The authors of (Li et al., 2021), for example, benefit from the attention mechanism to better proceed with the aggregation of features extracted by different layers.

Extending these methods to an ensemble of heterogeneous models would be relevant to exploit varied data, such as different representations of the same image or multispectral images. However, a strategy must be applied to properly combine features despite the differences in size and number of maps. In (Heller et al., 2022), we proposed a solution to reuse feature maps among heterogeneous architectures for a hierarchical approach. A PCA transformation efficiently addresses differences between the number of extracted maps and the number of maps expected by a specialized model, while a bilinear interpolation allows adjusting size differences. However, this solution was only proposed for a spanning tree architecture having a single feature extraction on which specialized models are grafted. Specialized models must process the same type of input data as the level 1 model, which contradicts our desire to exploit heterogeneous and complementary input data, such as multispectral images.

Many works focus on transferring knowledge from large and deep models to smaller models to achieve similar performances in a reduced computing time. Based on the popular KD approach, many studies use complementary labels to guide the training of a compact model. Feature-map-based methods seem to be more relevant because they allow the transfer of specific information, whereas label-based methods transmit information that is more abstract. However, the latter have the advantage of being able to use different architectures, whereas the former are often limited to similar or even identical networks. Some works have focused on knowledge transfer at the feature-map level among different architectures but are suitable for transfer from a single model. Feature aggregation methods are often applied on a single model or do not address heterogeneous data or architectures. It would therefore be

relevant to propose a multisource feature aggregation strategy that can work with an ensemble of models having heterogeneous architectures, either on different representations of the same image or on heterogeneous data such as multispectral images.

Both label-based and feature-map-based methods are aimed at making a compact model able to mimic the reasoning of a larger model by transferring various information. Their main objective is to extract the same features from the input data in a shorter time. In this paper, we propose another solution to mimic this reasoning—to reuse the feature maps—since we already have the models able to extract this information. Consistent with the work of (Heller et al., 2022), the overall model proposed here allows, however, to combine the feature maps extracted by different architectures to increase the performances. Compared to our previous work, several key changes have been made. First, instead of extracting the feature maps from a single model that processes the same type of input data, we extract them from multiple heterogeneous models, each one processing another type of data. Their fusion could, however, lead to accuracy inconsistency if we do not carefully consider the complementarity of the features. Second, and perhaps the most important, all the models in the proposed strategy do not process the same dataset of images. The literature shows us that multispectral images are relevant and that each spectral band or vegetation index is adapted for specific information. To best benefit from various information, it was compulsory to extend the solution to models that address different image types. For this purpose, the use of an ensemble of models is relevant, even if it implies proposing a solution to further reduce the computation time. Unlike classical ensemble solutions, the level 1 models do not need to be run in their entirety since we are only interested in the feature maps that they extract and not in their final output. We can thus benefit from various heterogeneous feature maps while running only a level 2 model in its entirety. Unlike existing approaches, we fuse features extracted with heterogeneous architectures exploiting different inputs. Table 1 summarizes the existing strategies, grouping them into main categories, with the main advantages and drawbacks compared to our proposed solution.

Our work is therefore also consistent with feature fusion for multispectral data. Substantial research has been performed on this subject. The main difficulty in such fusion is how to combine the information. The features utilized for the fusion can be extracted at different levels of a model; therefore, we can identify different types of fusion (Liu et al., 2016). According to the literature, middle fusion, which means using features extracted by intermediate layers, outperforms the other types of fusion. Most of the work, however, merges features from RGB and multispectral images only. For instance, the authors in (Qingyun & Zhaokui, 2022) propose a cross-modality attentive feature fusion strategy and infer attention maps from common and differential modalities. The features are, however, extracted by the same model from the different images. These solutions require images that are acquired under the same conditions and perfectly aligned. However, most of the time, there are shifts between two images, for example, due to the use of different cameras that are not in the same position. In these cases, image registration tasks are applied. For example, (Kerkech et al., 2020) applied such an approach for vine disease detection from UAV images by relying on key points obtained by the AKAZE algorithm (Alcantarilla et al., 2013). Identifying such key points is a complex and time-consuming task that can introduce artifacts, whereas our solution completely disregards this step. We can use images with parallax effects or recorded by sensors having different viewpoints, while most of the works use RGB and multispectral superposed images. Note that the proposed strategy can also handle different representations of the same image or any combination between RGB images and multispectral images. For this reason, the first application concerns the well-known PlantVillage dataset composed of RGB images, while the second application addresses multispectral images acquired with one and two multispectral cameras.

Table 1
Advantages and drawbacks of the main strategies compared to our proposed solution.

Solution	Advantages	Drawbacks
Multispectral imaging	Accuracy, first symptoms, Vegetation indices	Rarely used in real-world applications, sensitive to environmental conditions and parallax effects
Deep learning cumbersome models	Accuracy, generalization abilities	Needs large, labeled datasets for training
Label-based KD	Different architectures	Abstract information
Feature-maps-based KD	Specific information	Similar architectures or a single teacher
Feature aggregation	Different sources	Homogeneous networks
Classical grafting approaches	Heterogeneous networks	Single feature extraction, same type of data
Multispectral data fusion	Different sources	Same model, perfectly aligned images, image registration
Proposed solution	Accuracy, generalization abilities, different sources, heterogeneous networks, specific information, insensitive to parallax effects	Position of GN and complementarity between features empirically evaluated

3. Proposed method

In this paper, we propose to use an ensemble of level 1 models that could have heterogeneous architectures to extract feature maps that can be reinjected into a single compact level 2 model. During the inference stage, it is not practical to use an ensemble of models, even if the accuracy is better than with a single model. We therefore propose to apply our solution at low-level layers and only need an ensemble of models up to the layer where we apply an aggregation step during the inference stage.

The ensemble of models can also be utilized for hierarchical classification tasks at the first level of a hierarchy for the main problem, thus requiring the lightest architectures and shortest training times. The level 2 model will be employed in this case for the subproblem of the hierarchical classification task. We propose a solution that merges a hierarchical approach and ensemble learning strategy while requiring a single inference.

To be adapted to real-world applications, the level 2 model must be a compact architecture, possibly capable of running in real time and/or on an embedded system. As we will discuss in the application section, this model is not sensitive to the parallax effect that occurs with multisensor cameras and does not require a preprocessing step prior to the inference. The model can also process data from different sources, including those with different viewpoints.

Our proposed strategy is composed of four steps. First, we train an ensemble of level 1 models, which may not have similar architectures. However, each model optimizes the same main problem using a different input, e.g., classification of different plants from the PlantVillage dataset using a different color space of the same image. Our approach has significant flexibility since we can use various models and work with data from different sources. In this paper, we limit ourselves to images, but nothing prevents us from using data of different types (for instance, images and numerical data on weather conditions at the time of acquisition).

Second, we simply extract the feature maps for each level 1 model and remove all the subsequent layers by fixing a layer that may be different for each level 1 model. While this approach may seem counterintuitive, we do not need the subsequent layers or the final outputs from each level 1 model. This deletion saves important time at the inference stage as well as resources in the case of implementation on embedded systems.

Third, we choose a level 2 model that will be fed by a subset of the extracted feature maps. Since we have to respect the specific format of feature maps for this level 2 model, we will not use a classical architecture but a "cut_model" introduced by Heller et al. (2022), i.e., an architecture where the first layer applied is not a classical input layer but rather an intermediate layer. We usually build the cut_model based on a classical existing CNN architecture by removing its n first layers. This parameter n is, however, difficult to determine and is empirically selected here.

Last, once we know the expected input format of the cut_model, we apply a transformation block to each level 1 model to transform the selected feature maps to make them usable by the cut_model. The transformation block reduces the number of feature maps and changes their size. We propose in Section 3.4 two different transformation blocks to better control the complementarity of feature maps according to the outputs of the first step and the performances of each level 1 model.

Fig. 1 illustrates the global methodology on voluntarily generic data since the flexibility of the strategy makes it applicable to a very wide range of applications.

We can use different level 1 networks with different architectures to extract heterogeneous low-level feature maps. Thus, the variability of the features is twofold: due to the various models and due to the different data sources. In the latter case, we can use different representations in different color spaces of the same input RGB image, since the authors of (Gowda & Yuan, 2019) noted that it is relevant to use an ensemble of models on different representations of the same image. For instance, a first representation could be obtained by using the identity function, a second representation could be a conversion to the Lab color space, and a third representation could be a conversion to the HSV color space. Thus, even if we only have a single data source, the proposed solution is still relevant. The concept can be applied to any data, including different image sources, as we observe in the last two applications where multispectral images are utilized.

3.1. Ensemble model

The first step of our strategy is relatively simple and is presented in Fig. 2.

The objective here is to feed each model in the ensemble with a different input (a different version of the same input or a different image acquired by a different sensor) so that they can extract different features. This task is obviously not optimized if the different images are generated from the same base image since the inputs are still highly correlated. However, even those small changes may be enough to improve the complementarity of the features and move closer to the reasoning of an ensemble model. Naturally, the use of independent inputs gives better results, as we observe in the last two applications using multispectral images.

3.2. Level 1 models: feature extraction

In the second step, we fix the layer of each level 1 model from which we will extract feature maps, each one being composed of N_i maps of size L_i*l_i, while respecting certain constraints. We want each selected layer to output enough feature maps so that we can exploit an important amount of information from each level 1 model within the level 2 model. For these reasons, we always work with the output of convolutional layers.

Another point to consider is the individual performance of each model. Since we address different models, it is unlikely that all models will have the same performance. It can then be relevant to keep more features from the best models. This point will be further discussed in Section 3.4. This approach offers important flexibility since we can, in addition to fusing information extracted by different networks and on different data, perform multiscale fusion. This capability can be especially suitable if the different input data have multiple input sources, i.e., if they have been acquired with different sensors, for example. The selection of this layer is similar to that for the Grafting Node in (Heller et al., 2022) since we use the same type of level 2 model, but with a key difference. Here, the only role of the level 1 models is to extract features, whereas they also served as classifiers in (Heller et al., 2022). Thus, we will not need the layers located after the feature extraction point. Hence, it is relevant to place this point early in each of the models from the ensemble to delete an important number of layers, whereas it was relevant to place it deeper in the network in the initial grafting solution. Typically, the extracted feature maps that are combined represent the output of one of the first five convolutional layers of the various networks. One of the limitations of our approach is that the selection is empirical. It would be relevant to rely on more theoretical concepts to more robustly identify positions.

Once we have selected the position of each layer, we remove all the subsequent layers of each level 1 model since they will no longer be useful. Note that even if we keep only a few layers for each level 1 model, we still train them in a classical way to ensure the best possible features for the task at hand. Even if they are utilized only as feature extractors at the inference stage, the models from the ensemble are trained as classifiers.

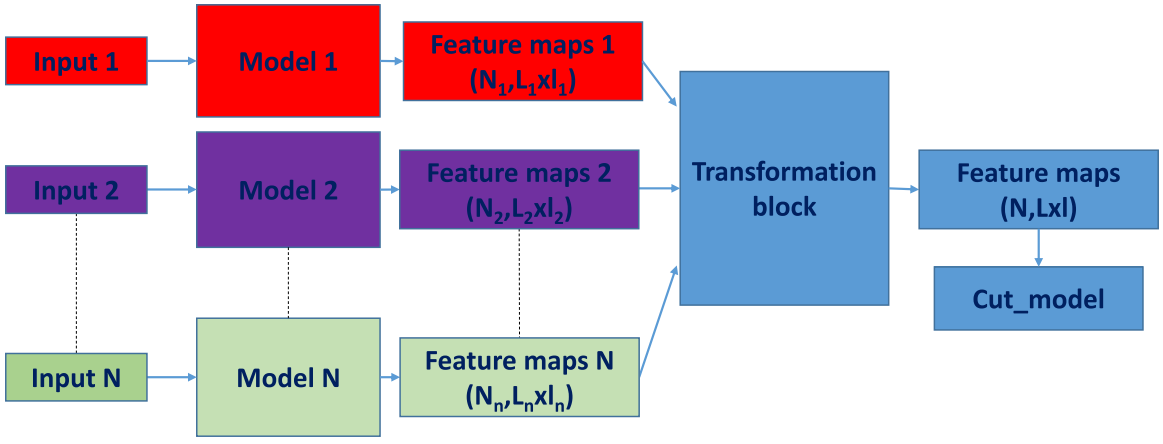


Fig. 1. Global methodology with an ensemble of N models applicable to any type of data. Each model from 1 to N is trained to classify between superclasses C_1 to C_n. The selected feature maps from each model are transformed and fused by the transformation block. The cut_model can be trained to classify between superclasses C_1 to C_n but also between classes C_{i1} to C_{in} of one superclass C_i. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

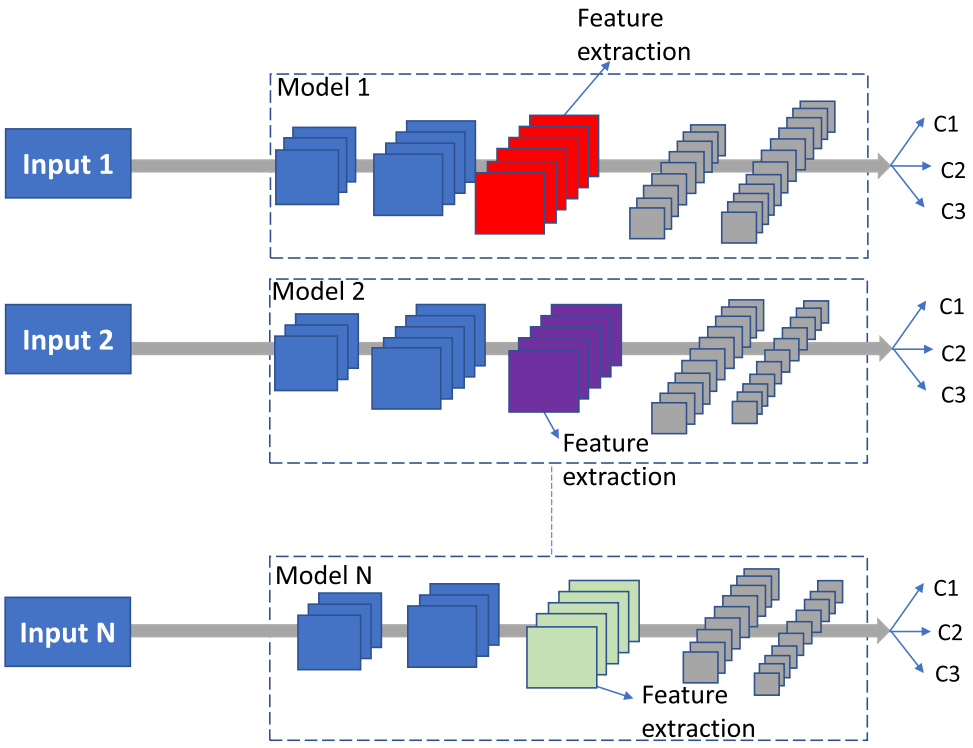


Fig. 2. Overview of the architecture of the ensemble model. Each model uses a different input and extracts low-level feature maps. The different feature maps are heterogeneous and can vary in size and number depending on the respective input. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

3.3. Level 2 model: feature fusion

Now that we have the ensemble of models, each one having few layers, we need to select a level 2 model that will receive the feature maps from all the level 1 models.

Let us assume that the level 2 model is trained on a subproblem, e.g., classification of diseases for one specific plant, and that a specific model achieves interesting results for this task. We decide to use this model as the level 2 model in our process. Since we already have a consequent number of feature maps, we will not use them as classical input data for the level 2 model but rather reinject them further in the model in a layer expecting N feature maps of size L*l. All the previous layers of this model are removed, defining what we refer to as a "cut_model", which saves additional time. To facilitate the next step, we prefer to start the level 2

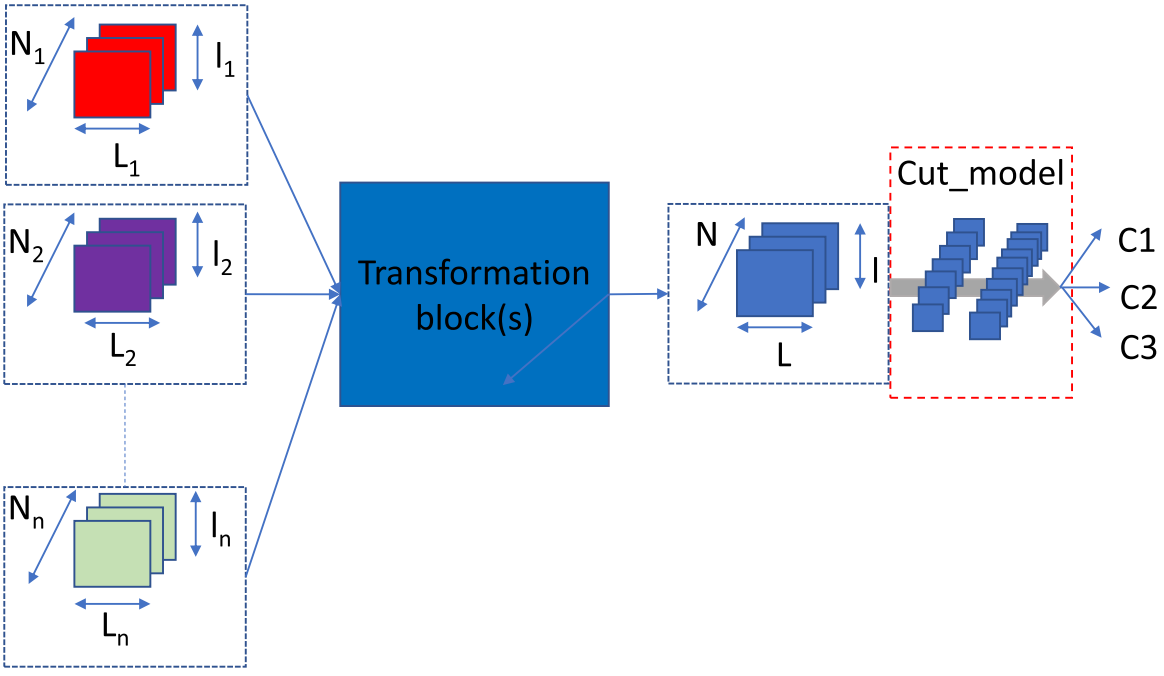


Fig. 3. The extracted low-level feature maps are resized and merged by a transformation block, resulting in a new set of feature maps that can feed the Cut_model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

model from a layer whose depth is quite similar to the layers of level 1 models in which we extract the features. For instance, in the case in which we use the same architectures for all the models and extract the features at layer N of the level 1 models, it would be relevant to start the cut_model directly at layer $N + 1$.

Fig. 3 illustrates a situation in which the cut_model expects an input dimension of N*L*l, and we extract N_i feature maps of size L_i*l_i from each level 1 network, with $\Sigma N_i > N$. To obtain the expected number and format of feature maps, we use one or several transformation block(s) that will be responsible for extracting the correct number of maps and resizing them to the expected size, from L_i*l_i to L*l, so we can concatenate them to be able to run the cut_model. Two different transformation blocks, "Global block" and "Independent block", are presented in Section 3.4. Unlike (Heller et al., 2022), where the proposed solution corresponds to a tree spanning hierarchy, the proposed method here represents a converging architecture in which all branches lead to the same compact model, the cut_model.

The cut_model is trained in a classical, supervised way for the problem it addresses, but with the output of the concatenation of the feature maps generated by the level 1 models as input. Depending on the problem, we can modify the number of maps that we obtain from each level 1 model to optimize the performance of the cut_model. Thus, the solution is relatively flexible and can be adapted to different problems. When we do not have a priori knowledge about the best feature maps that are extracted from the level 1 models, or when we want to ensure that we exploit the maximum information from each level 1 model, we can use the same amount of feature maps from each model. Conversely, if some level 1 models are more relevant than others, for instance, if they reach a significantly better accuracy, we can select most of the features from these models. The feature maps generated by the other level 1 models will therefore be utilized to make small adjustments and obtain small improvements. Even if they individually perform less well, the complementarity of the features generally leads to an improvement in the accuracy of the cut_model. It is, however, necessary not to have too many features from the same model. In this situation, the cut_model predictions will be very close to those of this individual model and will not benefit from the heterogeneity of data to perform more robust predictions. Worse, the other features can behave similar to noise, leading to accuracy inconsistency.

The cut_model can perform the same task as the ensemble of the level 1 models; for instance, all models are trained for a subproblem such as disease detection vs. normal plants. However, our concept can also be applied to hierarchical classification tasks. In this case, we can use the feature maps obtained for the main problem by training the level 1 models on this task, e.g., classification between two different plants, and change the architecture of the cut_model for a subproblem, e.g., disease detection. Thus, the solution is not only useful for optimizing the performance of the cut_model on a specific task for which the level 1 models are trained—plant classification in this example—but can also help increase the performance on other tasks, which are still linked to the main problem, such as identification of the diseases. This scenario is considered in the application section. We also determine that our method is particularly effective in reducing not only the number of parameters when the same task is performed by level 1 and level 2 models but also the processing time when we want to address a subproblem.

Due to the concatenation layer, we force the level 2 model to work with heterogeneous features and to move closer to the way an ensemble model would work at a faster speed. For the solution to be viable, an essential step is the transformation block.

3.4. Transformation block: feature transformation

Since one of our goals is to transfer knowledge between two architectures without needing to be similar, we must address three issues. First, we need to propose a solution to reduce the total number of feature maps without losing information. Second, we must be able to combine the feature maps despite their different dimensions (since we can use different architectures in the ensemble model). Last, it is compulsory to preserve the information from the different sources through the fusion process (since the data can derive from different sources).

Since we are using an ensemble of models, we will generate an immense number of feature maps, easily greater than that expected as the input of the level 2 model. Principal component analysis (PCA) can address this problem since it preserves the main information while reducing the number of feature maps. Note that more advanced solutions such as autoencoders might produce better results.

PCA, despite its simplicity, leads to impressive results while being fast. The autoencoders, for their part, can achieve slightly better accuracy but are substantially slower. We have two main concerns: benefitting from the ensemble models to improve the classification task and optimizing the process to be fast enough for real-world applications, possibly embedded on low-capacity devices. To gain better accuracy than achieved by PCA, we must use large or deep autoencoders, which means sacrificing the processing speed. Hence, if the accuracy is satisfactory using PCA, we use this dimension reduction method as a part of the transformation block.

After successfully obtaining the expected number of maps, we still must change their dimension to match the format expected by the cut_model. Moreover, to gather all the feature maps in the concatenation, they must have the same dimensions (only the number of maps can vary). The resizing of each map can be performed with bilinear interpolations, making it possible to increase or reduce their dimension as needed.

This transformation can also be achieved by using neural networks, possibly at the same time as the dimensional reduction. However, a simple bilinear interpolation is very fast to perform, and it seems disproportionate to use autoencoders to resize each feature map.

We must preserve the maximum amount of information from each input. Two situations can arise:

- The feature maps are compatible, which means that they have the same size, typically when we represent the same input image in different color spaces. In this case, first, we combine all the feature maps after the bilinear interpolation of each map and then run the PCA. Second, linear combinations are obtained with the features from each data and optimally executed. The feature maps from each model are resized, and the PCA will only be run once. This "global" transformation block is represented in Fig. 4.

If data are derived from different sources (different sensors, different cameras, or data of different types), we cannot directly apply PCA without losing information. Indeed, even a small shift between the input images will generate blurred areas that will be hardly exploitable. Without any preprocessing, we will lose essential information. An alternative solution is to use "Independent" transformations and apply the PCA on each set of feature maps before resizing them, as represented in Fig. 5. Thus, the input of the cut_model consists of a set of feature maps that have been built from a single input. We therefore ensure that each input data point is correctly represented. This solution also works with sensors having different viewpoints.

We apply the proposed method on the well-known PlantVillage dataset composed of RGB images and show that our approach is an innovative way to transfer knowledge. We then focus on multispectral images acquired with a DJI P4 Multispectral NIR (near-infrared) camera, with a small but regular parallax effect between the images acquired by the different sensors. Then, we add to these data other acquisitions from a multispectral SWIR (shortwave infrared) camera. This last application is more difficult since the parallax is not regular and the resolution of those images is also different due to the use of a different detector.

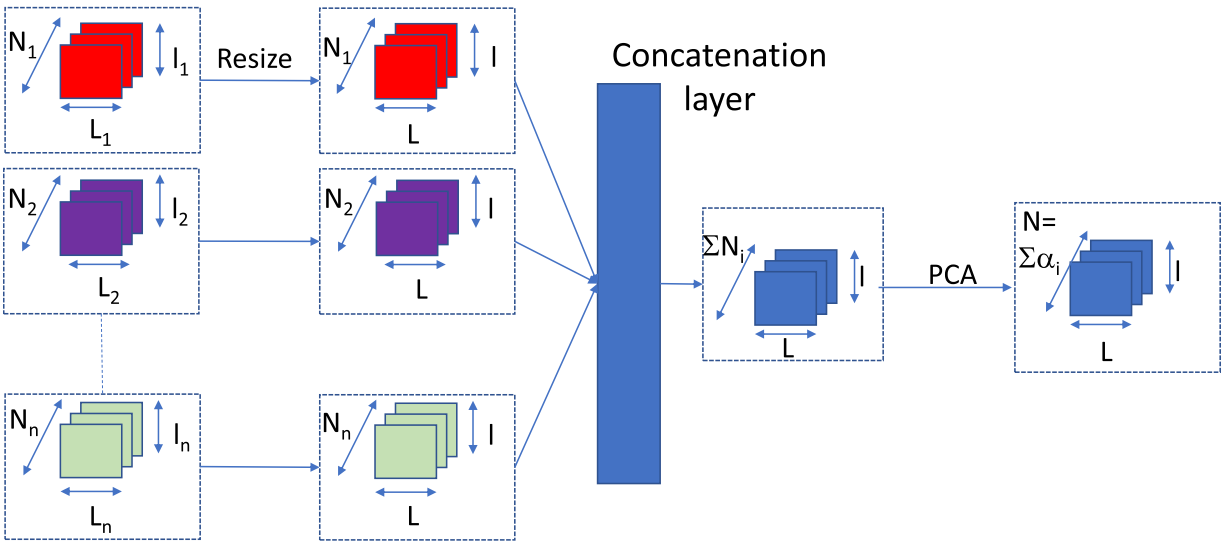


Fig. 4. "Global" transformation block: PCA is applied only once, after the maps have been concatenated, to build a combination from different sources. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

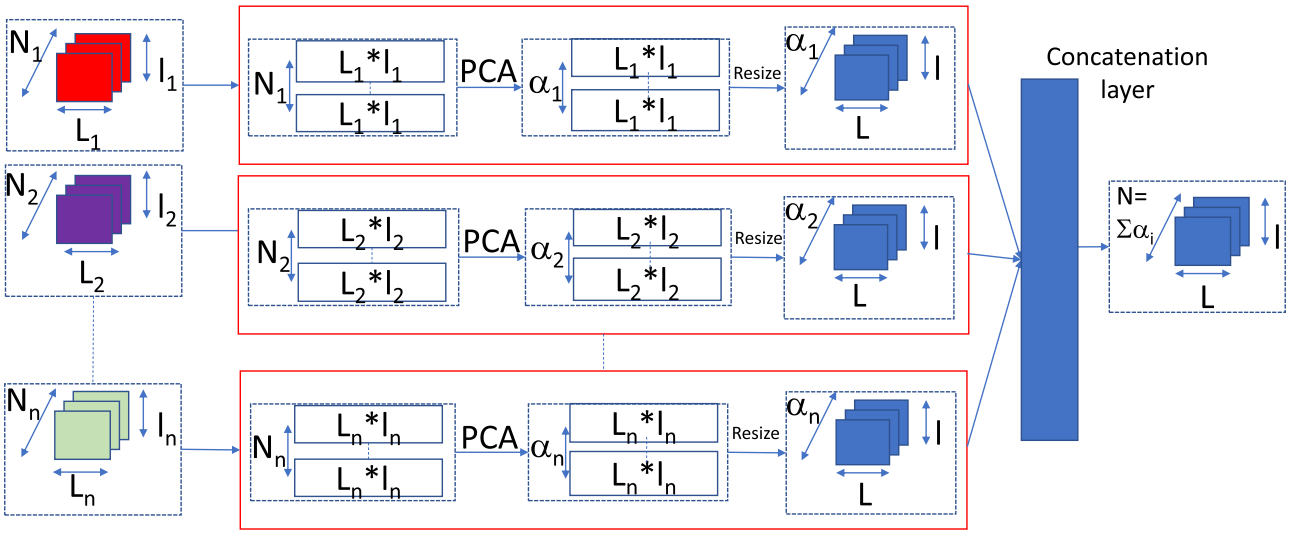


Fig. 5. "Independent" transformation block. PCA is applied to each set of feature maps to ensure that we retain information from each model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

4. Experiments

The objective of this section is to validate the performances of the proposed approach for classification tasks in realistic applications. We show that we always achieve a gain in accuracy compared to a classical model trained in a classical way. We compare the accuracy of our proposed method shown in Fig. 2 with each "full" version, named "Model i", i.e., the same architecture that was directly applied to the same input (image converted to the same color space or the respective spectral band). Compared to the diagram of Fig. 2, each "Model i" was trained on the respective input, and the accuracy was evaluated on the final output classes.

The speed tests were performed on a Dell precision 3530 (Intel Xeon E-2176 M CPU @ 2.70 GHz and NVIDIA Quadro P600) using Python 3.7.1, TensorFlow-GPU 1.14.0 and Keras 2.3.1. We ran each experiment five times and report the average results over these runs. Since the standard deviation was small and regular (± 0.1% in accuracy and F1 score for each model), we only indicate the average results.

The first two applications involve the PlantVillage dataset, which is a popular dataset of individual leaves of different plants infected with different diseases. For example, the dataset was used by Lee et al. (2020) to compare pretraining tasks on a global domain and specialized domain. The authors also suggested that it may be less relevant to train models in crop-disease terminology than to learn to distinguish the diseases independently of culture. We did not use the entire dataset but rather 6 different classes: four classes related to vine leaves (one healthy class and three different diseases) and two classes related to peach leaves (healthy and infected). The objective was to have classes containing symptoms (disease) that could be confusing on leaves with different shapes. Rather than adopting crop-disease terminology, we break down the classification task into two tasks: the main problem is to classify the crop, i.e., vine vs. peach leaves, and the subproblem is to classify the disease, i.e., healthy vs. each disease class. Thus, we aim to learn the discriminating symptoms of each disease and for each culture. We have a total of 6719 RGB images of size (256,256). To save time, each RGB image was employed as is and then converted to only two other color spaces: Lab and HSV. Each model was trained with the categorical cross-entropy loss and the Adam optimizer. The metric that we utilized

was the accuracy since the classes are relatively balanced. The initial learning rate was 1^e-3 and gradually decreased to 1^e-5. Different data augmentation options, such as rotations (from −45 to 45°), horizontal flip, zoom and translations, were randomly applied to the data. Gaussian noise and brightness variations were also applied with small probabilities.

The last two applications concern a custom dataset composed of 5 types of images, each acquired at a specific spectral band, as well as an RGB image recorded with a DJI P4 multispectral NIR camera. We thus have 6 representations of each leaf: blue (450 nm ± 16), green (560 nm ± 16), red (650 nm ± 16), red edge (730 nm ± 10), NIR (840 nm ± 26) and RGB acquired with a classical sensor. The dataset is composed of 2078 images per band. Each leaf represents one of 4 classes: healthy (624 images) and 3 different diseases, namely, Esca (155 images), Grapevine Yellow (775 images that have been identified after laboratory analysis by our Comité Champagne partner as Bois Noir, a disease that shares completely identical symptoms with Flavescence Dorée) and Grapevine leafroll-associated virus (524 images). These latter two classes shared many visual appearances, which makes classification difficult. It was interesting to use images of Bois Noir since the disease is much less epidemic than Flavescence Dorée but shares the same visual symptoms, which means that the model learns to identify the presence of one of these diseases within the Grapevine Yellow class, taking much less risk toward a vineyard. The application does not distinguish between Flavescence Dorée and Bois Noir, but the symptoms are so similar that it is not yet possible to distinguish them only by imaging techniques, and laboratory analyses are necessary. In this application, we are mainly interested in Grapevine yellow detection and thus consider the F1 score to evaluate our performances on the infected leaves at the first level and on the Grapevine Yellow class at the second level. We are mainly interested in the F1 score of the second level since we want to avoid false-negatives for the Grapevine Yellow class. However, the F1 score of the first level can also be relevant since it may indicate false negatives between instances of Grapevine yellow and healthy leaves that would occur during the initial classification.

For the last application, a second multispectral camera was utilized to make acquisitions on another 8 spectral bands in the SWIR domain, ranging from 900 to 1700 nm. We used a C-RED 3 camera built for us by the manufacturer First Light Imaging. The camera was equipped with a multispectral system based on a filter wheel containing 8 narrowband filters (10 nm) that we have chosen between 970 nm and 1650 nm.

Since we were particularly interested in the performances for the Grapevine Yellow class and classes are less balanced with more images for the healthy class, we applied the F1 score in addition to the accuracy.

4.1. Results on the Plantvillage dataset: all models trained for the same main problem

For our first experiment, we utilized EfficientNet-B0 as the architecture of the three level 1 models shown in Fig. 2 and selected another EfficientNet-B0 as the level 2 model to compose the cut_model shown in Fig. 3. These models can run in real time and achieve satisfying results on such classification tasks. We made this choice to better compare our approach but note that we could have used different architectures for each model. Here, the three models employed in the ensemble share the same architecture, with the variability in their representations due to changes in the input data (RGB, Lab, and HSV) and the random parameters applied during the training. We did not see any significant gain with models pretrained on ImageNet, so we preferred a random initialization to maximize the complementarity of features.

We start by training each of the level 1 models in its color space on the main problem. As the boundaries to separate peach and vine leaves are easy to identify, the training stage was sufficiently fast. Interestingly, this is the only time that we will need to train the level 1 models, which is a known limitation for ensemble techniques. The accuracy, speed, and number of parameters for each "Model i" on the respective color spaces are shown in the first 3 lines of Table 2.

For comparison purposes, we also considered more traditional approaches than our approach. The first approach, which we named "Fusion", consists of separately inferring the classes with each network and then making a decision by using a majority vote approach. The second approach, which we named "Softmax", consists of merging the last features extracted by the different level 1 models and then applying Softmax to perform a classification with these combined features. Note that both methods require that all three "Model i" models be integrally run.

To apply our approach, we chose the same position for the layers of each level 1 model, where we extracted the feature maps and the associated position to start our cut_model: layer 27 for the level 1 model and layer 28 for the cut_model. We made a second choice, i.e., layers 44 and 45, to evaluate the influence of the choice of an advanced layer. The cut_model, which is also based on an EfficientNet B0 architecture, was trained with the combined feature maps extracted from the different level 1 models, thus considering information from different spaces, but on the same main problem (classify peach vs. vine leaves). The choice of these levels was arbitrary, in the first case (levels 27–28) to have unspecialized features that can be quickly estimated and in the second case (levels 44–45) to have a more balanced overall architecture.

We observe that although we do not reach the results of the Softmax configuration (accuracy of 99.88%), we are still better when our approach is applied at an earlier layer (accuracy of 99.81%) than any of the individual models (best accuracy of 99.63%). The application to an advanced layer (levels 44–45) performed worse, perhaps due to less complementary features. As explained in Section 3, we prefer the first five convolutional layers to have a good balance between the quality of the features and the computation time. This reference supports our intuition since if we take a layer that is too advanced, in addition to wasting time, we negatively impact accuracy. However, as in our solution, we only use multiple models with few layers, and the processing time remains small, unlike when we fuse the three complete models. Note that the number of parameters also remains low, which is important in terms of the resources needed for implementation on embedded systems. Actually, we achieve an interesting balance since the accuracy of our approach is near the accuracy of the Softmax solution, which needs three complete models to run, is similar to a single model in terms of time and resources. From a speed point of view, our approach is slower than individual models but faster than ensemble strategies. Note that we can improve our processing time by using parallelization solutions to extract the features with the different level 1 models and thus achieve a speed that is closer to that of a single model. Here, we present the results without any implementation of such a strategy. The gain in accuracy is quite limited here since the individual models are already efficient. However, we achieve a gain in accuracy by applying color space transformations to a single image. This finding is encouraging for the following step, when we use more complementary features on a more difficult task.

Note that the results presented in Table 2 were obtained with the "independent" transformation block architecture, which means that we individually applied PCA on the feature maps from each model, as shown in Fig. 5, to preserve the data from each source. Once again, we

Table 2
Results obtained by using similar architectures (efficientnet-b0) for each level; models that were trained for the same task.

Model	Précision	Speed (FPS)	Parameters
RGB	99.63%	52.79	4,052,126
LAB	99.48%	52.79	4,052,126
HSV	99.48%	52.79	4,052,126
Ensemble	99.81%	17.55	12,156,378
Softmax	99.88%	17.39	12,156,374
Proposed solution (27–28)	99.81%	29.23	4,069,878
Proposed solution (44–45)	99.48%	26.2	4,099,650

can reduce the processing time by applying parallelization solutions to the different transformation blocks.

In the following step, we chose not to keep the same number of features for each model. Instead, we decided to keep a higher proportion of features for the RGB model (whose accuracy is 99.63% vs. 99.48% for the other models), i.e., $\alpha_1 > \alpha_2 = \alpha_3$. At this point, this decision had no impact on the accuracy, and we achieved the same results. It would be more relevant to do so for models with vastly different performances and distinctly superior models.

We also performed an additional test to determine whether we can replace a prior choice of the feature maps (choice of values of α_i) with a mechanism that will automatically choose the relevant combination of feature maps. We selected the global transformation block structure instead of the independent transformation block structure. First, we resize the individual maps so that they have the expected size. Second, we put the maps in the concatenation layer, obtaining an immense number of maps with the expected dimensions. We then converted each of them to a vector and applied a single PCA. This second possibility requires more computation resources than that proposed in Fig. 5 and is less suitable for parallelization solutions. It could, however, be adapted to cases where many level 1 models are employed. It, however, needs the data to be the same to avoid the blur phenomenon observed previously. The accuracy obtained in this case was 99.84%, the speed was 27.15 FPS, and the number of parameters is the same as shown in Table 2. We were also able to achieve better results than in the traditional supervised training (individual models), achieving slightly better accuracy similar to a Softmax ensemble model, but this approach is less flexible and slightly slower.

Although these results are interesting, all of the level 1 models share the same architecture. We therefore performed an additional test by replacing the RGB model with a MobileNet network and the HSV model with an EfficientNet B1 network, using one PCA per model ("Independent" transformation block proposed in Fig. 5). This new combination achieves an average result of 99.84%, which is slightly better than the accuracy result of 99.81% previously obtained with the same transformation block. This finding can be explained by a greater complementarity of features provided by the variety of architectures.

4.2. Results on the PlantVillage dataset: models trained for different problems

After having shown that our approach allows the use of complementary features to solve the main problem, i.e., classification between peach leaves and vine leaves, we then tested it within the framework of transferring information to solve a subproblem: classify diseases of vine leaves among themselves and diseases of peach leaves among themselves. We used the same configurations for the main problem, i.e., EfficientNet-B0 architecture for the three level 1 models but employed a cut_model based on the EfficientNet B3 architecture. This model is deeper since it is more difficult to distinguish two diseases on the same plant as two plants. The results of this second application are presented in Table 3. The "Best individual model" line corresponds to the performances obtained with a classical supervised approach by retaining the model that gives the best result among all the models, each being fed by Input i. The "Single level 1 model" is utilized to compare our performances with the particular case in which there is only one level 1 model fed by only one of the available inputs. This configuration is reminiscent of the grafting solution of (Heller et al., 2022). Indeed, if we do not consider the inference of the level 1 models, we consider the grafting solution as a particular case of our method, in which we only extract the features from a single model without searching for complementarity between two features. Here, we highlight our interest in using features from different sources rather than from just one source as cut_model inputs.

Once again, we achieved better results than the best classically model trained, which highlights the interest of the solution even for hierarchical classification tasks (remember that the level 1 models were not retrained for this new problem but only for the classification of plant species). In this situation, the accuracy is not affected when the solution is applied to a more advanced layer. A possible explanation is that the level 2 network is good enough to "hide" potential loss. We compare our solution to the particular case in which the ensemble of level 1 models is composed of a single model. The results show that it is relevant to use multiple level 1 models since we obtain better accuracy.

Table 3
Results obtained by using different architectures (EfficientNet-B0 and EfficientNet B3) for each level; models that were trained for different tasks: main problem for the level 1 models and sub-problem for the level 2 model.

	Model	Accuracy	FPS
Peach leaves	Best individual model	99.26%	26.9
	Single level 1 model	99.38%	26.9
	Proposed solution (13–14)	99.75%	26.01
	Proposed solution (27–28)	99.75%	26.28
	Proposed solution (three different networks, RGB only)	99.52%	25.6
Vine leaves	Best individual model	99.25%	26.9
	Single level 1 model	99.42%	26.9
	Proposed solution (13–14)	99.81%	26.01
	Proposed solution (27–28)	99.81%	26.28
	Proposed solution (three different networks, RGB only)	99.48%	25.6

To be exhaustive in our tests, we carried out the same experiment by replacing the three level 1 models, which had the same architecture (EfficientNet-B0), with three different models (EfficientNet B0, B1 and B2). Since we wanted to emphasize our interest of using different architectures, this time we only used RGB images as input instead of different color subspaces, i.e., each model was fed with the same RGB image. We still observed a gain in accuracy over traditional supervised learning, with an average accuracy of 99.48% for the vine leaves and an average accuracy of 99.52% for the peach leaves. The performance gain is less since we keep more feature redundancy, and the maps are all calculated from the same RGB image.

We conducted two other experiments for the classification of vine leaves: increasing the number of level 1 models and adding a multiscale aspect to the solution by selecting very distant candidates for each level 1 model (e.g., 5th, 20th and 40th convolutional layers). Adding other models designed to classify other color space images did not increase the accuracy. Worse, by adding one more model, the precision started to decrease, while the computing time increased. We might have introduced too much redundancy. The multiscale approach gave similar results, from a slight decrease (approximately 0.2%) to equal precision. The results of the literature, however, lead us to believe that it is possible to improve performance using this approach, with a less arbitrary choice of candidates.

Next, we apply our proposed method to our custom Grapevine Yellow dataset containing multispectral images. The following experiment shows that the solution is suitable for addressing multispectral data without any preprocessing, despite the parallax effect. Unlike the PlantVillage Dataset applications, we have data from different sources. Thus, we have less redundancy among the features, which should positively impact the accuracy.

4.3. Grapevine yellow detection: fusion of multispectral feature maps from a single multispectral camera with multiple sensors

We have already shown that our proposed strategy can benefit from the use of heterogeneous architectures applied to the same data, possibly represented in different ways. In this experiment, we show that even with the same base model, multispectral imaging can lead to an important gain in accuracy. For this reason, we chose a model identical to that utilized in the first experiment (Section 4.1), with the only exception being the inputs, which consisted of multispectral images

instead of different representations of the same RGB image. We therefore employed the same EfficientNetB0 architecture for each of the 5 spectral bands as well as for the RGB image. The cut_model architecture is also based on the EfficientNet B0 architecture. Additionally, we considered the same layers, i.e., layer 27 for the level 1 model and layer 28 for the cut_model.

However, in contrast to the previous application, we cannot apply the global transformation block without observing blurred areas. To do this, it would have been necessary to preprocess the input images by realigning them. The shift between the different images acquired for the 5 spectral bands is regular for the dataset that we acquired in the laboratory since it corresponds to a parallax shift due to the position of the sensors of the DJI P4 multispectral, NIR camera and can therefore be corrected. However, it is much more complicated for in situ acquisitions since parameters such as the distance from the object and the acquisition angle must be considered. Fortunately, the independent transformation block structure proposed in Fig. 5 makes it possible to combine the feature maps even though the images are not perfectly aligned. Avoiding the preprocessing step is another way to save calculation time.

We have addressed this classification task in two stages: first, we performed a binary classification between healthy leaves and contaminated leaves (3 different diseases). All models were trained on the main problem. The results are presented in Table 4. In this application, we are particularly interested in the detection of the Grapevine yellow, and above all, we want to avoid false negatives. It is therefore relevant to consider the F1-Score, including the healthy/infected status. As shown, the blue band carries less information than the other bands. Once again, we outperform the best individual models. Note that a bad model or bad input data can lead to accuracy loss, as shown with the fusion of 5 bands, which performs worse than the fusion of the 4 best bands. This loss is even more important when we consider the F1 score, which means that we perform worse on infected leaves because of less complementarity among the features.

Our proposed model succeeded in achieving better results than the best individual models despite the shift among the feature maps, which confirms that we do not have to preprocess the images to correct alignment issues. Another interesting point is that even with a larger ensemble, the processing time remains correct since we do not add too much computation for each model. We observe a loss of 1 to 1.5 FPS each time we add a level 1 model. This loss is mainly attributed to the computation of the feature maps and will only be observed the first time we process an image (for the healthy-infected status here). If we reuse the same feature maps within another cut_model to identify other diseases for the infected leaves, for instance, there is no significant difference, from a processing time point of view, among the configurations. We achieved an average gain of almost 0.5% compared to the RGB model, which is significant at these accuracy levels. This gain is slightly greater if we only consider the infected leaves, as highlighted by the F1 score. As expected, multispectral imaging is relevant for this type of application.

Second, once we classify the leaves, which was the main problem, we applied once again our solution, this time training the cut_model on a subproblem that classifies the diseases among them. As previously noted, we train each level 1 model on the respective images for the respective spectral band on the main problem to make a fair comparison, and our fusion solution is still based on the models that determine the health status. The difference stems from the level 2 model, which has the same architecture as previously noted (EfficientNet B0) but was trained with the feature maps extracted by the level 1 models as inputs instead of the initial input data. The results are presented in Table 5.

The results are quite similar to those of the previous experiment: the blue band is considerably the less informative, and we achieved better results than all the individual models and the RGB model. Since the task is more difficult, the gain achieved by our solution is more significant, with an F1 score for the Grapevine Yellow class increased by 1% compared to the RGB model and by 0.4% compared to the Red Edge model, i.e., the best individual model. Fig. 6(c) gives an example of a vine leaf infected by Grapevine yellow that our solution correctly classifies but that is classified as Grapevine leafroll-associated virus by the best individual models. This example can be visually compared to true positive leaves for both classes (Figs. 6(a) and 6 (b)).

This application confirms that we can optimize a subproblem of the task handled by the ensemble of networks. This finding is interesting since it means that even if we add new subclasses, for instance, we will not need to train once again the level 1 models but only a new cut_model. Although we did not focus on the training phase, this finding is still interesting, as it is a known limitation of methods based on multiple networks. The shift between the images implied by different positions of the sensors did not perturb our solution, and we successfully fuse feature maps from multispectral data.

We conduct further analysis by adding other images acquired with another camera. Indeed, even if we faced a parallax effect, it was a regular phenomenon that could be learned during the training phase of the cut_model (especially for the laboratory data). By adding a second camera, we ensure that our model can be used to process data from different sources.

4.4. *Grapevine yellow detection: fusion of multispectral feature maps from two different multispectral cameras*

For this second experiment, due to some difficulties during the acquisition phase of the SWIR images, we have fewer available images than those composing the multispectral NIR dataset. We were able to keep 1176 exploitable images per band from the previous NIR dataset. The shift between the SWIR image and the NIR image is not regular due to the acquisition conditions (the same leaves were placed at different positions to make the acquisitions), reproducing in situ acquisition conditions.

We individually evaluated the performances of the eight SWIR bands and observed that we achieved lower accuracy rates than using the NIR bands. We resized the data from (224, 224) to (300, 300) and used a

Table 4
Results obtained for the identification of the healthy/infected status by using similar architectures (EfficientNet-B0) for each level; models that were trained for the same task.

Data	Accuracy	F1 Score	FPS
RGB	99.28%	**99.12%**	26.9
Blue (B)	96.88%	96.5%	26.9
Green (G)	99.33%	99.33%	26.9
Red (R)	**99.55%**	**99.55%**	26.9
Red Edge (RE)	98.54%	98.66%	26.9
NIR	98.66%	98.66%	26.9
Proposed solution using 3 bands (G-RE-NIR)	99.58%	99.35%	26.28
Proposed solution using 4 bands (G-R-RE-NIR)	**99.76%**	**99.76%**	25.3
Proposed solution using 5 bands (B-G-R-RE-NIR)	99.33%	99.12%	24.2

Table 5
Results obtained for the identification of the disease by using similar architectures (EfficientNet-B0) for each level; models that were trained for different tasks: main problem for level 1 models and sub-problem for level 2 model.

Data	Précision	F1 Score Grapevine Yellow
RGB	**98.97%**	**98.81%**
Blue (B)	91.35%	90.82%
Green (G)	98.71%	98.81%
Red (R)	99.03%	99.03%
Red Edge (RE)	**99.35%**	**99.35%**
NIR	98.38%	98.54%
Proposed solution (G-R)	99.58%	99.35%
Proposed solution (G-R-NIR)	**99.76%**	**99.76%**
Proposed solution (G-R-RE-NIR)	**99.76%**	**99.76%**
Proposed solution (R-RE-NIR)	98.9%	98.81%

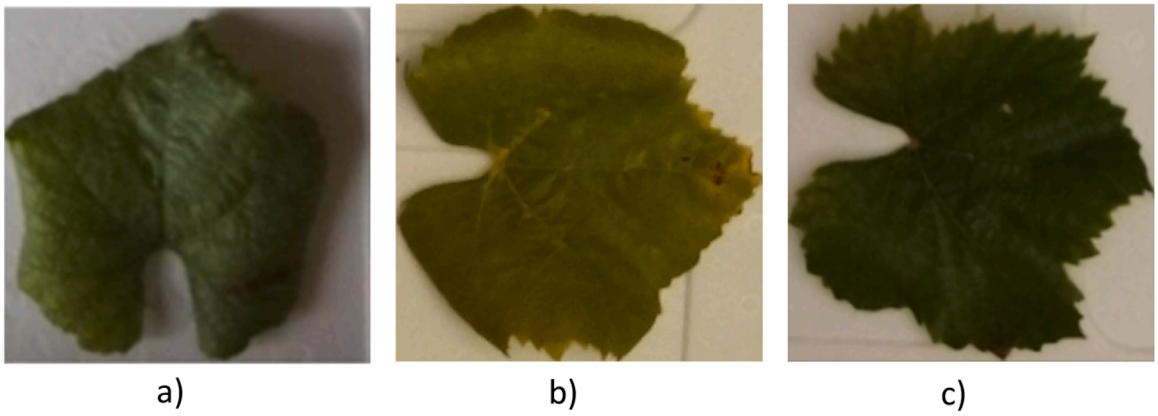


Fig. 6. (a) Grapevine leafroll-associated virus leaf correctly classified by all models; (b) Grapevine yellow leaf correctly classified by all models; (c) Grapevine yellow leaf that fools the best individual model being classified as Grapevine leafroll-associated virus but is correctly classified as Grapevine yellow by our solution. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article)

deeper architecture to try to improve our level 1 models on the two best bands (EfficientNet-B0 and EfficientNet-B3). The results are reported in Table 6. This drop in accuracy is mainly attributed to the least amount of training data. Indeed, by retraining the level 1 models for the NIR bands on the same training dataset, we achieved an accuracy of 93.83% for the green band, which was the best score and close to the best SWIR results.

We found that the most relevant fusion occurred between SWIR bands 2 and 6, and we achieved an accuracy of 93.6% (we are again seeing a gain in accuracy). We then performed a more complex fusion between the two selected SWIR models, the Green model (which achieved an accuracy of 93.83%) and the Red Edge model (which achieved an accuracy of 91.78%). As previously mentioned, here, we need to apply the independent transformation block since the band cannot be properly combined directly without preprocessing. With this fusion, we record the best result with an accuracy of 94.52% versus 93.83% for the best individual model. We confirm that the more difficult the classification task is, the greater the gain in precision achieved by our fusion strategy. We also confirm that without any preprocessing, we were able to combine data from two different cameras. Finally, the different resolutions were not problematic which is interesting because we can increase the resolution of the inputs that are more difficult to analyze but are relevant, such as NIR images, to gain accuracy while reducing the resolution of the images that are easier to process, such as RGB data, to gain speed.

4.5. Discussion

We proposed an independent transformation block that starts with PCA followed by resizing to be able to choose which feature maps will be concatenated. We also proposed another possibility, a global transformation block, by applying a single PCA after the concatenation of feature maps to let the proposed approach choose the relevant combination of feature maps.

When separately extracting the features of each model due to the independent transformation block, the computation time could be reduced if we change the size of the feature maps and then reduce their dimensions. This configuration will be privileged when l_i (L_i) is larger than l (L); therefore, we need to reduce the size of each map since PCA would be run faster on maps with smaller dimensions.

When we applied the independent transformation block, we employed the same number of maps from each level 1 model if possible, i.e., if the number of models is a divisor of the expected number of maps. In this situation, the α_i coefficients were equal. When these conditions are not met, or in other situations, one may choose larger coefficients for the best individual models. For example, one may want to force a model to work with features from different spectral bands while knowing that the NIR carries more relevant information. In this case, we can use more feature maps from this band than from the other bands (for example, $\alpha_{NIR} = 0.7*N$). The various α_i coefficients were empirical in this work according to the observed performances, which is another limitation of the proposed approach. In future work, it would be relevant to theorize this outcome to ensure that sufficient information is available from each source.

Note that the global transformation block allows bypassing the difficulty of properly choosing the coefficients α_i (the number of feature maps selected from each level 1 model) as the PCA builds the new maps with linear combinations of maps extracted from all level 1 models, but with the need for similar data or a preprocessing step. However, when we combined different architectures, if one of them outputs many more feature maps than the other (for instance, by three or four times), this transformation block will tend to give too much importance to these maps, and the other blocks will be almost interpreted as noise. We concluded that it is better if there are large differences in the number of feature maps extracted from each level 1 model, to force the variety of feature maps by favoring the use of one PCA per model, i.e., using the independent transformation block. Such a transformation can also be easily utilized with data of different types since there is no need to preprocess them to keep relevant information from each, which means that they can be more independent. Notably, we favored the latter solution as soon as the data was obtained from different sources, as shown by the last two experiments.

We showed that our model can be applied when the level 2 model shares the same architecture with the level 1 models but also has a different architecture. We have also noted that the complementarity of features could be artificially increased by applying simple yet effective modifications to the RGB input images. It could be relevant to increase the variety of features by using, in addition to these modifications,

Table 6
Results for each SWIR band for the disease identification problem. We identified the best bands by reproducing the same test for each of the 8 bands and then improved the performance on the two best bands with deeper models.

Spectral band	Model	Size	Accuracy
0	EfficientNet B0	(224,224)	87.02%
1	EfficientNet B0	(224,224)	83.17%
2	**EfficientNet B0**	**(224,224)**	**89.42%**
2	**EfficientNet B3**	**(300,300)**	**93.15%**
3	EfficientNet B0	(224,224)	87.02%
4	EfficientNet B0	(224,224)	87.02%
5	EfficientNet B0	(224,224)	87.98%
6	**EfficientNet B0**	**(224,224)**	**89.42%**
6	**EfficientNet B3**	**(300,300)**	**90.41%**
7	EfficientNet B0	(224,224)	88.94%

concepts such as attention.

The gains observed after applying modifications to the same data were quite limited because of the correlation among the features. However, when we used multispectral data, with less redundancy of information among the bands, and when the classification task was more difficult, we observed larger gains. The use of multimodal data, with nonimage data for instance, could be addressed by our method with independent block structures. Compared to classical ensemble solutions, we observe a consequent gain of time and computational resources due to the level 1 models that are neither utilized nor saved in memory in their entirety but only up to a layer. Therefore, the shallower the chosen layer, the greater the interest of the solution.

Even if the optimization of the position follows objectives that contrast with the grafting solution, we are still bound to the same main constraint, which is that we do not yet have a theoretical rule to determine an optimal, or at least a relevant, position. Hence, it can be necessary to try different combinations of feature maps before finding the best combination, which implies building and training different cut_models at the training phase, which can be penalizing in terms of computation time. In future work, it would be relevant to focus on information theory approaches to optimize, or even automatize, the selection of layer positions and to estimate the complementarity between two sources. Such approaches should reduce the empirical aspect of the proposed solution.

We skipped the preprocessing step of realigning images to save time, which led us to eliminate the vegetation indices that need perfect alignment among bands. If we find that such indices are relevant to use, nothing prevents us from realigning the images, constructing relevant indices and introducing them to our solution, with a model adapted to each index. We therefore respect the conditions to use a single PCA for all models if this is relevant.

5. Conclusions

We proposed an innovative feature aggregation method that uses different architectures, allowing us to overcome the limitations of existing knowledge distillation techniques when applied to an ensemble of teachers. In our solution, we train an ensemble of level 1 models on a problem but only use low-level feature maps that are transformed and concatenated due to PCA and interpolation techniques to feed a level 2 model. In this way, we can reuse the information on another compact model that will be trained on these feature maps by starting at an adequate level. As we work at the feature map level, our approach reduces the processing time and computational resources by reinjecting the information at a layer that is not the first of the level 2 models by simply removing all its previous layers. Further tests will focus on determining in a less empirical way the optimal layers for the extraction and reinjection of features, especially in the context of multiscale feature aggregation.

By forcing the level 2 model to work with features extracted from different models, i.e., from different representations of the same image or from data acquired by different sensors, we push it to have a functioning close to that of an ensemble model while remaining a compact model, thus observing a gain in accuracy compared to classical supervised training.

We succeeded in building a compact global model that can jointly process features from different sources, notably multispectral data. These results are even more interesting since no preprocessing is necessary to realign the input data. Not only is it not necessary to manage the parallax effect among multispectral images, but there is nothing to prevent us from using different data as long as we can extract feature maps. This approach enables many possibilities since neural networks can be applied to many types of data. A future step will be to apply the strategy to the real data acquired in vineyards.

From an application point of view, we achieved very good identification rates of Grapevine yellow disease in laboratory conditions (uniform lighting and same geometry of the acquisition system) by using multispectral images in our feature aggregation method. The application prospects for real data are encouraging. Indeed, we can use the most relevant bands to perform the classification by compensating them for their sensitivity to environmental conditions using less affected bands. It will now be necessary to apply this solution to in situ acquisitions. The method could be of additional interest under these conditions, allowing the relevance of multispectral data to be applied while providing additional robustness to variations in experimental conditions with other less sensitive input data.

Another interesting aspect of the proposed solution is that it can be applied to hierarchical classification tasks, successfully reusing feature maps for a subproblem that is not that handled by the ensemble model. The saving of time compared to ensemble models is important here since the feature maps have already been extracted. We even manage to achieve a computation time similar to that of a single classical network.

We also proposed two possibilities to apply a solution depending on whether we want the repartition of the feature maps to be manually or automatically determined. The latter requires more resources and achieves slightly better results. However, the level 1 models must process the same type of data (same dimensions and perfectly aligned images). The independent block structure is much more flexible.

The solution is designed to work with architectures that are different, and we obtain relevant results when we transfer knowledge between two different EfficientNetB0, from an EfficientNetB0 to an EfficientNetB3, and from three different models.

To our knowledge, none of the existing work thus far has been carried out to reuse the feature maps from an ensemble of models with different architectures and from multiple sources to a compact model. Further tests will be carried out with other architectures. We will also seek to determine the best combinations of spectral bands to optimize the results of classification for in situ conditions and extend the solution to other types of data.

One of the main interests of our solution is that it can be applied to a large panel of applications, even beyond the smart agriculture domain, since we only need different representations of the input data, which can derive from specific cameras with different sensors and different cameras but can also be artificially simulated via classical colorspace changes. It has indeed been proven many times in the literature that ensemble solutions can achieve gains in accuracy for many applications at the cost of increasing the computation time. In this paper, we proposed a solution to benefit from this accuracy gain without adding too many computations, therefore improving the accuracy/speed balance of such solutions. Furthermore, we showed that we can use an ensemble of models with different architectures, which is another advantage for the solution to be applied in various applications. The proposed solution can be applied as it is, with possible adaptations for on-board computing, to process images acquired by drones. An interesting improvement would be to combine images acquired from different viewpoints. However, even if we can combine images without realigning them, it will probably be more difficult to combine images taken from very different angles. It will then probably be necessary to separate the information flows in the cut_model and process several “batches” of feature maps rather than combining incompatible maps. Projects are currently being investigated with the Champagne committee to define a test framework in which the proposed solution, as well as other solutions developed by manufacturers, can be tested in interaction with winegrowers and champagne committee experts.

CRediT authorship contribution statement

Guillaume Heller: Conceptualization, Methodology, Software, Writing – original draft. **Eric Perrin:** Conceptualization, Methodology, Writing – review & editing. **Valeriu Vrabie:** Conceptualization, Methodology, Writing – review & editing. **Cedric Dusart:** Validation, Supervision. **Marie-Laure Panon:** Data curation. **Marie Loyaux:** Data

curation. **Solen Le Roux:** Supervision.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Guillaume Heller reports financial support was provided by National Association of Technical Research. Guillaume Heller reports a relationship with Segula Technologies that includes: employment. Guillaume Heller reports a relationship with Reims Champagne-Ardenne University that includes: employment.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the Association Nationale de la Recherche et de la Technologie (ANRT).

References

Ahmad, A., Saraswat, D., & El Gamal, A (2023). A survey on using deep learning techniques for plant disease diagnosis and recommendations for development of appropriate tools. *Smart Agricultural Technology, 3*, Article 100083. https://doi.org/10.1016/j.atech.2022.100083

Albetis de la Cruz, J.L. (2018). *Potentiel des images multispectrales acquises par drone dans la détection des zones infectées par la flavescence dorée de la vigne* [These de doctorat, Toulouse 3]. https://www.theses.fr/2018TOU30157.

Alcantarilla, P., Nuevo, J., & Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Procedings of the British Machine Vision Conference 2013*. https://doi.org/10.5244/C.27.13, 13.1-13.11.

Al-Saddik, H., Simon, J. C., Brousse, O., & Cointault, F. (2017). Multispectral band selection for imaging sensor design for vineyard disease detection: Case of Flavescence Dorée. *Advances in Animal Biosciences, 8*(2), 150–155. https://doi.org/10.1017/S2040470017000802

Asif, U., Tang, J., & Harrer, S. (2020). Ensemble knowledge distillation for learning improved and efficient networks (arXiv:1909.08097). arXiv. https://doi.org/10.48550/arXiv.1909.08097.

Ba, L.J., & Caruana, R. (2014). *Do Deep Nets Really Need to be Deep?* (arXiv:1312.6184). arXiv. https://doi.org/10.48550/arXiv.1312.6184.

Blalock, D., Ortiz, J.J.G., Frankle, J., & Guttag, J. (2020). *What is the state of neural network pruning?* (arXiv:2003.03033). arXiv. https://doi.org/10.48550/arXiv.2003.03033.

Boulent, J., St-Charles, P.-L., Foucher, S., & Théau, J. (2020). Automatic detection of flavescence dorée symptoms across white grapevine varieties using deep learning. *Frontiers in Artificial Intelligence, 3*, Article 564878. https://doi.org/10.3389/frai.2020.564878

Bucila, C., Caruana, R., & Niculescu-Mizil, A. (2006). *Model Compression, 2006*, 535–541. https://doi.org/10.1145/1150402.1150464

Caruana, R., Niculescu-Mizil, A., Crew, G., & Ksikes, A. (2004). Ensemble selection from libraries of models. In *Twenty-First International Conference on Machine Learning - ICML '04*. https://doi.org/10.1145/1015330.1015432, 18.

Coulibaly, S., Kamsu-Foguem, B., Kamissoko, D., & Traore, D. (2022). Deep learning for precision agriculture: A bibliometric analysis. *Intelligent Systems with Applications, 16*, Article 200102. https://doi.org/10.1016/j.iswa.2022.200102

Furlanello, T., Lipton, Z.C., Tschannen, M., Itti, L., & Anandkumar, A. (2018). *Born Again Neural Networks* (arXiv:1805.04770). arXiv. https://doi.org/10.48550/arXiv.1805.04770.

Gong, Y., Wang, L., Guo, R., & Lazebnik, S. (2014). *Multi-scale orderless pooling of deep convolutional activation features* (arXiv:1403.1840). arXiv. https://doi.org/10.48550/arXiv.1403.1840.

Gowda, S.N., & Yuan, C. (2019). *ColorNet: investigating the importance of color spaces for image classification*. 11364, 581–596. https://doi.org/10.1007/978-3-030-20870-7_36.

Heller, G., Perrin, E., Vrabie, V., Dusart, C., & Le Roux, S. (2022). Grafting heterogeneous neural networks for a hierarchical object classification. *IEEE Access : Practical Innovations, Open Solutions, 10*, 12927–12940. https://doi.org/10.1109/ACCESS.2022.3144579

Heo, B., Lee, M., Yun, S., & Choi, J. Y. (2019). Knowledge transfer via distillation of activation boundaries formed by hidden neurons. *Proceedings of the AAAI Conference on Artificial Intelligence, 33*(01), Article 01. https://doi.org/10.1609/aaai.v33i01.33013779

Hinton, G., Vinyals, O., & Dean, J. (2015). *Distilling the knowledge in a neural network* (arXiv:1503.02531). arXiv. https://doi.org/10.48550/arXiv.1503.02531.

Ji, M., Heo, B., & Park, S. (2021). *Show, attend and distill:knowledge distillation via attention-based feature matching* (arXiv:2102.02973). arXiv. https://doi.org/10.48550/arXiv.2102.02973.

Kerkech, M., Hafiane, A., & Canals, R. (2020). Vine disease detection in UAV multispectral images using optimized image registration and deep learning segmentation approach. *Computers and Electronics in Agriculture, 174*, Article 105446. https://doi.org/10.1016/j.compag.2020.105446

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. https://www.semanticscholar.org/paper/Learning-Multiple-Layers-of-Features-from-Tiny-Krizhevsky/5d90f06bb70a0a3dced62413346235c02b1aa086.

Kundu, N., Rani, G., Dhaka, V. S., Gupta, K., Nayaka, S. C., Vocaturo, E., & Zumpano, E. (2022). Disease detection, severity prediction, and crop loss estimation in MaizeCrop using deep learning. *Artificial Intelligence in Agriculture, 6*, 276–291. https://doi.org/10.1016/j.aiia.2022.11.002

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06), 2*, 2169–2178. https://doi.org/10.1109/CVPR.2006.68

Lee, S. H., Goëau, H., Bonnet, P., & Joly, A. (2020). New perspectives on plant disease characterization based on deep learning. *Computers and Electronics in Agriculture, 170*, Article 105220. https://doi.org/10.1016/j.compag.2020.105220

Li, M., Lei, L., Tang, Y., Sun, Y., & Kuang, G. (2021). An attention-guided multilayer feature aggregation network for remote sensing image scene classification. *Remote Sensing, 13*(16), Article 16. https://doi.org/10.3390/rs13163113

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2022). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems, 33*(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Liu, J., Zhang, S., Wang, S., & Metaxas, D.N. (2016). *Multispectral deep neural networks for pedestrian detection* (arXiv:1611.02644). arXiv. https://doi.org/10.48550/arXiv.1611.02644.

Park, S., & Kwak, N. (2019). FEED: Feature-level ensemble for knowledge distillation. ArXiv:1909.10754 *[Cs]*. http://arxiv.org/abs/1909.10754.

Qingyun, F., & Zhaokui, W. (2022). Cross-modality attentive feature fusion for object detection in multispectral remote sensing imagery. *Pattern Recognition, 130*, Article 108786. https://doi.org/10.1016/j.patcog.2022.108786

Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., & Bengio, Y. (2015). *FitNets: hints for thin deep nets* (arXiv:1412.6550). arXiv. https://doi.org/10.48550/arXiv.1412.6550.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision, 115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Tan, M., & Le, Q.V. (2020). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks* (arXiv:1905.11946). arXiv. https://doi.org/10.48550/arXiv.1905.11946.

Urban, G., Geras, K.J., Kahou, S.E., Aslan, O., Wang, S., Caruana, R., Mohamed, A., Philipose, M., & Richardson, M. (2017). *Do deep convolutional nets really need to be deep and convolutional?* (arXiv:1603.05691). arXiv. https://doi.org/10.48550/arXiv.1603.05691.

Wang, S., Celebi, M. E., Zhang, Y.-D., Yu, X., Lu, S., Yao, X., Zhou, Q., Miguel, M.-G., Tian, Y., Gorriz, J. M., & Tyukin, I. (2021). Advances in data preprocessing for biomedical data fusion: An overview of the methods, challenges, and prospects. *Information Fusion, 76*, 376–421. https://doi.org/10.1016/j.inffus.2021.07.001

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2016). *Show, attend and tell: neural image caption generation with visual attention* (arXiv:1502.03044). arXiv. https://doi.org/10.48550/arXiv.1502.03044.

Yang, Y., Lv, H., & Chen, N. (2023). A Survey on ensemble learning under the era of deep learning. *Artificial Intelligence Review, 56*(6), 5545–5589. https://doi.org/10.1007/s10462-022-10283-5

Zagoruyko, S., & Komodakis, N. (2017). *Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer* (arXiv:1612.03928). arXiv. https://doi.org/10.48550/arXiv.1612.03928.

Zhang, Y., Deng, L., Zhu, H., Wang, W., Ren, Z., Zhou, Q., Lu, S., Sun, S., Zhu, Z., Gorriz, J. M., & Wang, S. (2023). Deep learning in food category recognition. *Information Fusion, 98*, Article 101859. https://doi.org/10.1016/j.inffus.2023.101859

Zhang, Y.-D., Dong, Z., Wang, S.-H., Yu, X., Yao, X., Zhou, Q., Hu, H., Li, M., Jiménez-Mesa, C., Ramirez, J., Martinez, F. J., & Gorriz, J. M. (2020). Advances in multimodal data fusion in neuroimaging: Overview, challenges, and novel orientation. *Information Fusion, 64*, 149–187. https://doi.org/10.1016/j.inffus.2020.07.006