Trends in Plant Science

CellPress
REVIEWS

Feature Review

Deep Learning for Plant Stress Phenotyping: Trends and Future Perspectives

Asheesh Kumar Singh,[1] Baskar Ganapathysubramanian,[2] Soumik Sarkar,[2,*] and Arti Singh[1,*]

Deep learning (DL), a subset of machine learning approaches, has emerged as a versatile tool to assimilate large amounts of heterogeneous data and provide reliable predictions of complex and uncertain phenomena. These tools are increasingly being used by the plant science community to make sense of the large datasets now regularly collected via high-throughput phenotyping and genotyping. We review recent work where DL principles have been utilized for digital image–based plant stress phenotyping. We provide a comparative assessment of DL tools against other existing techniques, with respect to decision accuracy, data size requirement, and applicability in various scenarios. Finally, we outline several avenues of research leveraging current and future DL tools in plant science.

Highlights

Review of DL techniques applied to plant stress (biotic and abiotic) phenotyping to drive transformational changes in agricultural sciences.

Comparative assessment of DL strategies across a wide range of plant species for plant stress identification, classification, quantification, and prediction (ICQP), specifically focusing on digital image–based phenotyping.

Best practices, future avenues, and potential applications of DL techniques in plant sciences with a focus on plant stress phenotyping, including deployment of DL tools, image data fusion at multiple scales to enable accurate and reliable plant stress ICQP, and use of novel strategies to circumvent the need for accurately labeled data for training the DL tools.

Deep Learning

Recently, we reported on the potential and possibilities of utilizing machine learning (ML) for high-throughput stress phenotyping in plants [1]. With the rapidly increasing sophistication, capability, and miniaturization of imaging sensors, the plant science community is facing a data deluge of plant images under various environments and under various stresses (biotic and abiotic). This ability to perform high-throughput phenotyping has resulted in increasing interest in automated approaches to extract features (i.e., symptoms and organs) of physiological interest from these large datasets with the intent of identifying and quantifying plant stresses. We complement our earlier review by focusing specifically on a very promising and rapidly advancing subset of ML tools in this work: deep learning (DL). This is especially important and topical due to the remarkable advances in DL tools that have transformed several disciplines, including consumer analytics, autonomous vehicles, automated medical diagnostics, and automated financial management. This is also an area that is quickly becoming the workhorse strategy for most ML applications (Figure 1 compares ML papers with DL papers over a 10-year period from 2008 to 2018). Our goal in this review is to provide a comprehensive overview, infer trends, and identify outstanding problems that the plant science community could pursue as we integrate DL concepts into our domain. We limit our review to DL applications that primarily utilize image data. This is motivated by the fact that digital imaging is relatively cheap; can be deployed in a scalable manner; can be easily integrated with manual, ground, and aerial platforms; and requires potentially the least technical expertise to deploy with off-the-shelf components for high-throughput plant phenotyping. This has resulted in a veritable explosion of research activities using digital imaging for plant applications.

Machine Learning in Plant Science

ML (and hence DL) concepts can be deployed on four broad categories of problems in plant stress phenotyping [1]. These categories form part of the so-called 'ICQP' paradigm with the acronym representing the four categories (i) identification, (ii) classification, (iii) quantification,

[1]Department of Agronomy, Iowa State University, Ames, IA, USA
[2]Department of Mechanical Engineering, Iowa State University, Ames, IA, USA

*Correspondence:
soumiks@iastate.edu (S. Sarkar) and
arti@iastate.edu (A. Singh).

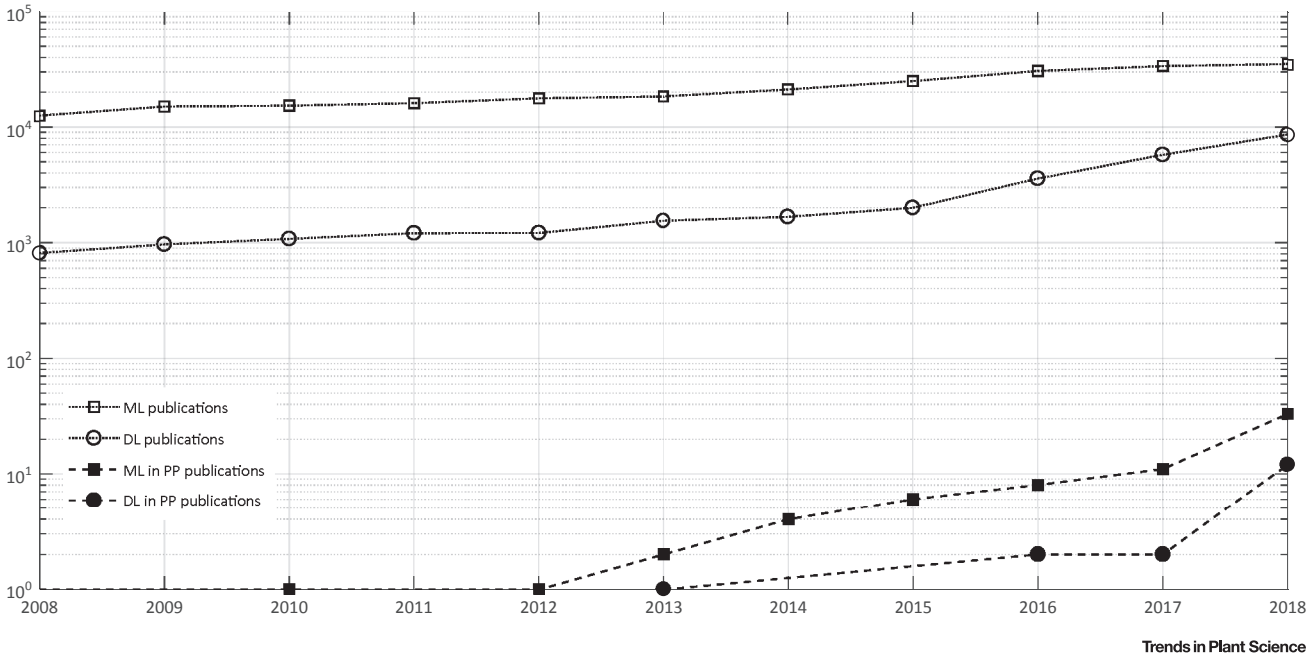


Figure 1. Comparative Assessment of Publications Related to Machine Learning and Deep Learning. The open symbols indicate all topics, while the closed symbols indicate plant phenotyping topics. Notice the log scale for the y axis. Source: Web of Science with keyword search 'Machine Learning'; 'Deep Learning'; 'Machine Learning and Plant Phenotyping'; and 'Deep Learning and Plant Phenotyping' using the 2008–2018 period. DL; deep learning; ML; machine learning; PP; plant phenotyping.

and (iv) prediction. These four categories naturally fall into a continuum of feature extraction where increasingly more information is inferred from a given image. Identification refers to detection of specific stress, that is, simply identifying which stress is being exhibited, for example, sudden death syndrome in soybean or rust in wheat. Classification is the next step, where ML is used to classify the image on the basis of stress symptoms and signatures. Here, the goal is to place the visual data (leaf, plant, canopy, or row) into a distinct stress class (e.g., low-, medium-, or high-stress categories). Quantification involves a more quantitative characterization of stress, such as incidence and severity. Disease incidence is defined as the rate of new cases of the disease, which is reported as the number of cases occurring within a period of time or at any time instant (generally at the time of maximum disease expression). In plant pathology, a common way to describe disease incidence is the percentage of diseased leaves on a single plant or the number of diseased plants out of the total number of plants in a field or plot [2]. Disease severity is a more detailed quantification measure and is reported as the area of plant tissue affected by the disease (commonly presented as a percentage) on a leaf or on the entire plant canopy [2]. The last category is prediction of plant stress ahead of time, before visible stress symptoms appear. This has substantial implications for the timely and cost-effective control of stress and is one of the key drivers of precision and prescriptive agriculture.

Machine Learning Approaches for Reliability and Accuracy

In plant stress phenotyping a useful plant stress assessment strategy must satisfy the following requirements. (i) Reliability: the degree to which measurements of the same diseased individuals obtained under different conditions produce similar results [3,4]. Reliability relates the magnitude of the measurement error in observed measurements to the inherent variability

in the 'true' sample. The two types of reliability in plant disease assessment are intra- and interrater reliability. Intrarater reliability is the agreement between assessment (trait measurement) by the same rater taken temporally, while interrater reliability is agreement between assessment of the same plot or sample taken by different raters. (ii) Accuracy: the closeness of an estimate to the actual value. This is heavily influenced by the rater experience level. It is now well understood that these requirements are met with automated phenotyping workflows that enhance both reliability and accuracy by reducing bias (e.g., use of images rather than rater subjectivity), by removing rater fatigue (use of automated phenotyping systems), and by correct feature extraction (currently done using ML tools). This has spurred the development of various tools like the iPad app 'Estimate' [5], a smartphone app for sugar beet disease detection [6], and the smartphone app 'Leaf Doctor' [7]. These tools seek to improve disease rating data quality by decreasing human error and to some extent inter- and intrarater variation. While conventional image processing has proved useful, the wide variability in quality and complexity (i.e., occlusion, debris, changes in illumination intensity and shading, and loss of function) of the images make consistent application of standard image processing strategies challenging to the ICQP paradigm. This is where ML (and especially DL) tools enable the creation of reliable workflows for feature identification.

ML enables algorithmic learning from experience. According to Arthur Lee Samuel, ML is defined as the 'field of study that gives computers the ability to learn without being explicitly programmed'. Here, we distinguish between ML techniques that use 'handcrafted features' (i.e., *a priori* user-identified features) that are used for ICQP [1] versus the more recent and promising DL techniques that do not require any hand-crafting of features, as they are able to automatically 'learn' the features or representations from the image data (Figure 2). In ML, feature hand-crafting refers to the exercise of choosing appropriate parts (e.g., one color channel from an RGB image) or transformations [e.g., scale invariant feature transform (SIFT)] that is applied to the raw datasets before training to enhance the performance of an ML model. While ML experts have developed different procedures of feature extraction depending on data and model types, they are still predominantly heuristic. Therefore, the feature extraction process in traditional ML involves time-consuming trial-and-error steps and success may depend on the level of experience of the data scientist. In this regard, one of the fundamental advantages of DL is that it involves an automatic hierarchical feature extraction process via learning a large bank of nonlinear filters prior to performing decision-making, such as classification. Hence, DL models typically work quite well with raw data and do not require the trial-and-error–based hand-crafted feature extraction process.

Note that this applies to all ML problems regardless whether they are supervised (i.e., learning from data with target labels) or unsupervised (i.e., learning from data without target labels). While using traditional ML, both supervised and unsupervised learning approaches typically require feature extraction for better performance. For example, for large dimensional data, both support vector machine (SVM, a supervised technique) and K-means clustering (an unsupervised technique) may benefit from a principal component analysis (PCA)-based feature extraction (i.e., performing PCA and selecting only few top principal components for learning). On the other hand, a deep convolutional neural network (CNN, a supervised model) or a deep autoencoder (an unsupervised model) may not need any such feature extraction and can leverage the raw data directly.

Our focus in this review is on the applicability and promise of DL tools as they have shown exceptional success in recent years on complicated phenotyping problems with good predictive ability. We hope that the plant science community can leverage the rapid advances and

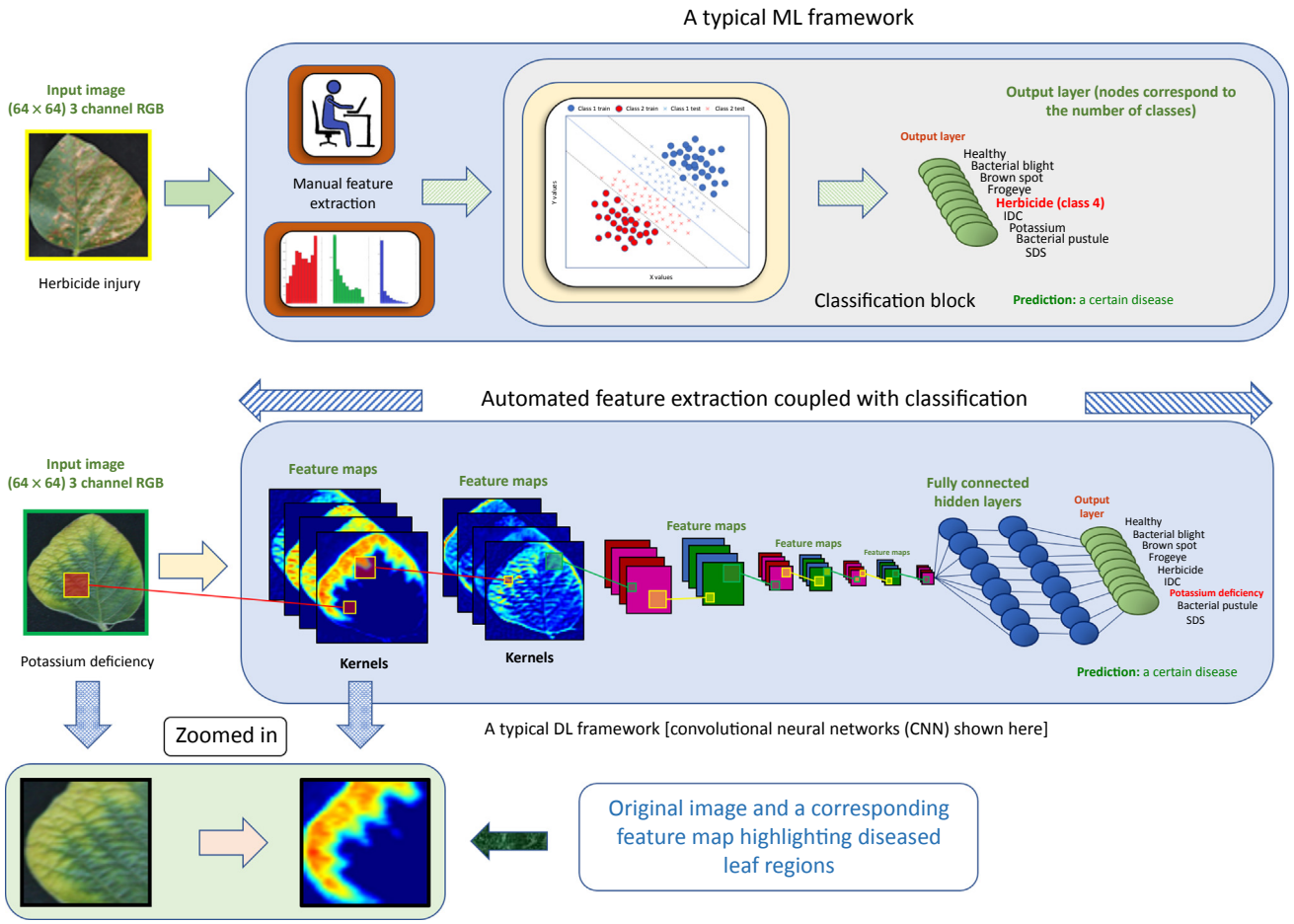
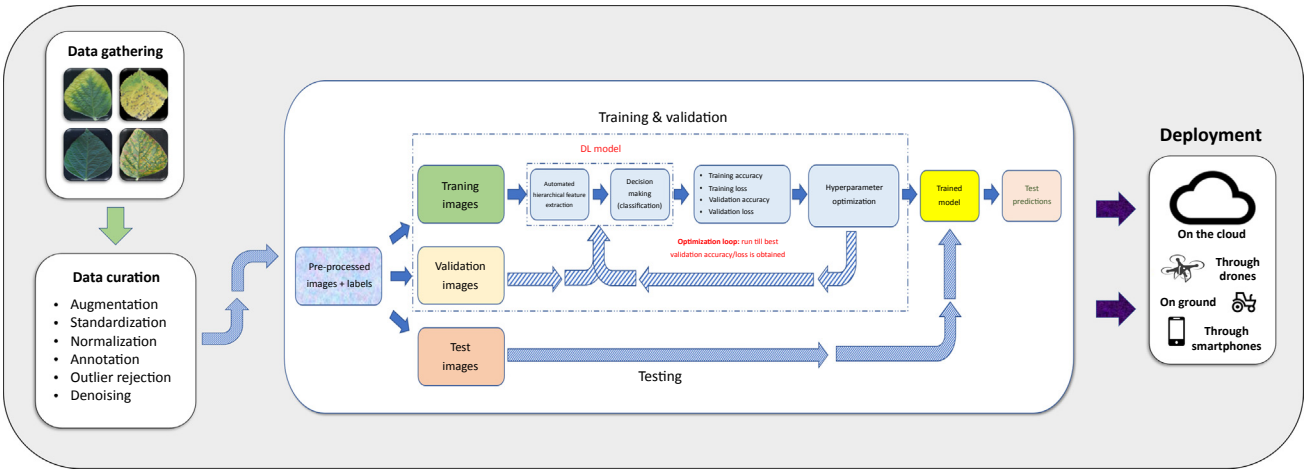


Figure 2. Key Differences between Machine Learning (ML) and Deep Learning (DL) Paradigms. An ML pipeline typically consists of two stages. Stage A: a human user identifies key features to extract from image data, either based on intuition, domain expertise, or hypothesis. These features are then extracted from images, either manually or using image-processing algorithms. Stage B: this feature set is then used in subsequent classification/regression analysis. In contrast to this two-stage approach, a DL pipeline is typically fed the raw image data and, during training, automatically extracts features that result in the best classification/regression. The advantage of a DL pipeline is that no feature identification (or feature engineering) is needed. The disadvantage of a DL pipeline is that the ensuing automatically identified features may be difficult to interpret physiologically.

application of DL tools (from successful non-agricultural applications) to enable transformative advances in agriculture. We next introduce the basic idea of DL.

The Basics of Deep Learning

DL is a class of ML techniques which utilizes a stack of multiple processing layers where each succeeding layer uses the output from the previous layer as input to learn representations of data with multiple levels of abstraction (Figure 3). Typically, DL models are built using multilayer neural networks where two subsequent layers of features are connected by neurons that essentially represent various parametrized nonlinear transformations. Examples of model parameters include weights and biases of the neurons that get multiplied and added to the input, respectively. In the forward direction, input data gets transformed in a layer-by-layer fashion until the target layer or the decision layer. For example, for an image classification problem, an input image is transformed using the hierarchical nonlinear transformations and



Trends in Plant Science

Figure 3. An Illustration of a Deep Learning (DL) Tool Chain from Data Gathering to Decision Making. A key first step is to gather a large, diverse set of data. The data is then curated. Curation includes standardization, outlier rejection, some simple denoising and image preprocessing, and data augmentation. This dataset is then split into training, validation, and testing subsets. The training and validation subsets are used in training the DL architecture/network. Training the DL network essentially means optimizing the parameters (weights, biases) of the network such that the network accurately learns the mapping from the input data to the desired output label. The trained model is then finally tested on the unseen test image data subset. After successful completion of this testing, the DL network can be deployed for inference. While training the DL model can be resource intensive, deploying a trained model is relatively simple.

finally the image class becomes the output at the target layer. Training such a model begins with an initial parameter set (often randomly chosen) for the deep neural network (DNN). Errors are computed at the target layer between the actual outputs and the desired outputs given by the training data labels for a large number of examples. Then the errors are used in a feedback mechanism in a layer-by-layer fashion (from target to input) to update the parameters until a satisfactory level of decision accuracy is achieved at the target layer. Typically, a method called the error back-propagation algorithm is used for this training process. The typical features learnt by a DNN are hierarchical in nature, that is, while initial layers capture low-complexity fundamental features (such as edges and corners for an image classification problem), more complex features are formed at the higher layers of abstraction via complex combinations of the low-complexity features. Stochastic gradient descent (SGD) and its variants, such as minibatch gradient descent [8], ADAM [9], and ADMM [10], have been used to train DNNs.

A Brief History of Deep Learning

The foundation of DL started in the mid-1960s when Ivakhnenko and Lapa used multiple layers of nonlinear features with polynomial activation functions in an approach similar to DL as we know it today [11]. The next significant milestone was the use of neural networks [12]; however, weights were manually assigned. This was followed by application of back-propagation of errors to train deep models that could yield useful distributed representation. Then during the 1990s, the concept of CNNs was introduced [13] specifically for image recognition problems. However, the power and possibilities of DL remained unrealized primarily because of three reasons: (i) lack of very large datasets, (ii) lack of computing power, and (iii) certain algorithmic deficiencies. These issues started to get resolved in the mid-2000s with the advent of 'big data', which provided very large datasets and graphics processing units (GPUs) that provided computing resources and algorithmic advancements. Neural network capabilities started to be appreciated with these models showing significantly better performance than traditional ML models for most of the benchmark classification and prediction problems. In a seminal paper

[14], Hinton and Salakhutdinov demonstrated the impressive capability of DNNs, which opened the door for a wide variety of uses of DL. Pioneering DL development and implementation studies [15–17] specifically on deep convolutional neural network (DCNN) in the areas of speech processing and image processing attracted technological giants, such as Google, Facebook, Amazon, NVIDIA, and Microsoft, to invest significant resources in DL development. We see the impact of such investment in our everyday lives in the form of smartphone apps and consumer microtargeting. While traditional ML and computer vision techniques retain a role in specific instances (where large datasets are not available or there is a computation constraint), many current and future impactful scientific and technological innovations are becoming a reality via leveraging DL concepts.

Training Deep Learning Models

DL models can be trained in both supervised and unsupervised ways. In supervised DL, labeled input data (such as an image of a diseased leaf) are mapped to output (e.g., a soybean disease such as sudden death syndrome) via a weights vector and errors are back-propagated (adjusting the weights) from the output layer to the input; whereas, in the case of unsupervised DL, the objective is to identify patterns from the data for various purposes, such as clustering and hashing. A key difference between traditional supervised ML and supervised DL is that DL combines the two-step process of feature extraction and decision-making of traditional ML within one model and avoids the often suboptimal manual handcrafting (Figure 2). As discussed earlier, training DL models typically involve SGD, an iterative optimization algorithm (or variants thereof, such as ADAM), to accomplish the back-propagation-based (to update weights) model parameter learning. However, the choice of hyperparameters in the training process determines (to a large extent) a successful DNN model. Important hyperparameters include the network architecture (such as the number of units in a layer and the number of layers), the learning rate, and the choice of activation functions. While network architecture can be chosen carefully for an individual problem at hand, it is common practice in the community to begin with a preselected architecture that has been shown to be successful in various data domains and problems and adapt it to the problem under consideration. Examples of such popular architectures include AlexNet [17], ZFNet [18], VGGNet [19], InceptionNet [20], ExceptionNet [21], and ResNet [22], which are briefly introduced in the next section. The process of leveraging such pre-existing network architectures effectively is called ‘transfer learning’ and is discussed in a later section (Table 1).

Apart from hyperparameter choices for DL training, another key aspect that requires attention during training such models is the issue of overfitting, when the model is learning excessive details about the training data (such as modeling the noise in training data), thus leading to significantly poor performance on the unseen test data (i.e., when the model is deployed). Before the resurgence of artificial neural networks (ANNs) in the form of DL models, the lack of approaches to prevent overfitting was one of the key reasons ANNs became unpopular in the ML community. Although large DL models with often millions of learnable parameters can suffer from similar issues, use of very large datasets has substantially reduced such problems. Therefore, if a sufficiently large dataset is not available, it is often very useful to *augment* the dataset (primarily by various transformations of original images, such as rotation). Apart from data augmentation and traditional regularization techniques to avoid overfitting, novel regularization techniques such as dropout [23] have emerged to reduce overfitting DL models. Another algorithmic modification called batch normalization also helps deep network training by avoiding the problem of covariate shift [24]. We refer the interested reader to comprehensive reviews, one that focuses on the theory [8] as well as easy-to-follow tutorial that provides a gentle introduction to practitioners [25].

Table 1. Examples of Deep Learning Approaches in Plant Stress Image-Based Phenotyping

DL algorithm application (ICQP)	DL algorithm type	Plant	Platform	Stress type	Stress name	Refs
Identification	LeNet architecture	Banana	Manual	Biotic stress	Early scorch, cottony mold, ashen mold, late scorch, tiny whiteness	[67]
Identification	AlexNet, GoogLeNet, VGGNet-16, ResNet-20	Apple	Manual	Biotic stress	*Alternaria* leaf spot, mosaic, rust, brown spot	[68]
Identification	Inception-v3, ImageNet	Cassava	Manual	Biotic stress	Cassava brown streak disease, cassava mosaic disease, brown leaf spot, cassava green mite damage, cassava red mite damage	[69]
Identification	AlexNet, ALexNetOWTBn, GoogLeNet, Overfeat, VGG	Apple, banana, blueberry, cabbage, cantaloupe, cassava, celery, cherry, corn, cucumber, eggplant, gourd, grape, onion, orange	Manual	Biotic stress	Bacterial spot, apple scab, cedar apple rust, black rot, banana sigatoka, banana speckle, brown leaf spot, cassava green spider mite, *Cercospora* leaf spot, common rust, northern leaf blight, esca (black measles, late and early blight, cucumber mosaic, downy mildew, powdery mildew, frogeye leaf spot, leaf scorch, *Septoria* leaf spot, *Septoria* leaf blight, spider mites, tomato mosaic virus, leaf mold, target spot, TYLCV, huanglongbing	[66]
Identification	AlexNet, GoogLeNet	Apple, blueberry, cherry, corn, grape, peach, bell pepper, potato, raspberry, soybean, squash, strawberry, tomato	Manual	Biotic stress	Apple scab, apple black rot, apple cedar rust, cherry powdery mildew, corn gray leaf spot, corn common rust, corn northern leaf blight, grape black rot, grape black measles, grape leaf blight, orange huanglongbing (citrus greening), peach bacterial spot, bell pepper bacterial spot, potato early blight, potato late blight, squash powdery mildew, strawberry leaf scorch, tomato bacterial spot, tomato early blight, tomato late blight, tomato leaf mold, tomato *Septoria* leaf spot, tomato two-spotted spider mite, tomato target spot, tomato mosaic virus, tomato yellow leaf curl virus	[53]
Identification	Modified LeNet	Olive	Manual	Biotic stress	Olive quick decline syndrome	[71]
Identification	CNN	Cucumber	Manual	Biotic stress	Melon yellow spot virus, zucchini yellow mosaic virus, cucurbit chlorotic yellows virus, cucumber mosaic virus, papaya ring spot virus,	[72]

Table 1. (continued)

DL algorithm application (ICQP)	DL algorithm type	Plant	Platform	Stress type	Stress name	Refs
					watermelon mosaic virus, green mottle mosaic virus	
Identification	CaffeNet, ImageNet	Pear, cherry peach, apple, grapevine	Manual	Biotic stress	Porosity (pear, cherry, peach), powdery mildew (peach), peach leaf curl, fire blight (apple, pear), apple scab, powdery mildew (apple), rust (apple, pear), grey leaf spot (pear), wilt (grapevine), mites (grapevine), downy mildew (grapevine), powdery mildew (grapevine)	[88]
Identification	AlexNet, ZFNet, VGG-16, GoogLeNet, ResNet-50, ResNet-101, ResNetXt-101, Faster RCNN, R-FCN, SSD	Tomato	Manual	Biotic and abiotic stress	Gray mold, canker, leaf mold, plague, leaf miner, whitefly, low temperature, nutritional excess or deficiency, powdery mildew	[52]
Identification	CNN	Maize	UAV	Biotic stress	Northern corn leaf blight	[75]
Identification	VGG-FCN, VGG-CNN	Wheat	Manual	Biotic stress	Powdery mildew, smut, black chaff, stripe rust, leaf blotch, leaf rust	[73]
Identification	VGG-A, CNN	Radish	UAV	Biotic stress	*Fusarium* wilt	[74]
Identification	SCRNN	Tomato	Manual	Biotic stress	Bacterial leaf spot, early blight tomato, late blight, *Septoria* leaf spot, two-spotted spider mite, tomato mosaic virus, tomato leaf mold, target spot of tomato, and tomato yellow leaf curl virus.	[70]
Classification	AlexNet, GoogLeNet	Tomato	Manual	Biotic stress	Tomato yellow leaf curl virus, tomato mosaic virus, target spot, spider mites, *Septoria* spot, leaf mold, late blight, early blight, bacterial spot	[77]
Identification, classification, quantification	AlexNet	Soybean	Manual	Biotic and abiotic stress	Bacterial blight, bacterial pustule, frogeye leaf spot, *Septoria* brown spot, sudden death syndrome, iron deficiency chlorosis, potassium deficiency, herbicide injury	[81]
Quantification	VGG-16, VGG-19, Inception-v3, ResNet50	Apple	Manual	Biotic stress	Black rot	[78]
Prediction	DNN	Tomato	Manual	Abiotic stress	Water stress	[84]

Popular Deep Learning Architectures

Convolutional Neural Networks

CNN architectures are leveraged in many applications. Major CNN architectures for image recognition include AlexNet, ZFNet, GoogLeNet [26], VGGNet, and ResNet (see the

supplemental information online); more recently, hybrid models using Inception and ResNet called Inception-ResNet, Xception network, and DenseNet have been introduced. In addition to image recognition, extremely useful DCNN architectures were proposed for simultaneous objection detection and localization. Some of the most popular of those architectures are Region CNN (RCNN) [27]; Fast RCNN [28,29]; Faster RCNN [30]; Mask RCNN [31], which combines the faster RCNN and the fully convolution network (FCN) [32]; single shot multibox detector (SSD) [33]; and You Only Look Once (YOLO) [34,35].

Summaries of major CNN architectures origins and capabilities can be seen in the supplemental information online.

Unsupervised Deep Neural Networks

While supervised DNNs have been immensely successful, in many cases a large volume of labeled data is not available or there may be a need for informative feature extraction from data without specific targeted decision-making (such as recovering signal from noisy data). For these applications, unsupervised DNNs can be quite useful. Deep belief networks (DBNs) [36], which are built layer-by-layer using restricted Boltzmann machines (RBMs), are one of the early unsupervised deep models that have been widely used [37]. The other most popular type of unsupervised deep model is called a deep autoencoder [38]; its variants have been extensively used for many important applications such as denoising, in-painting, and hashing [39].

Deep Neural Networks for Time Series

While deep CNNs are predominantly used for spatial data, deep recurrent neural networks (RNNs) are built for modeling sequential data such as time series [40,41]. However, standard deep RNNs may not be quite suitable for modeling time series with long-range order or memory. In such cases, deep long short-term memory (LSTM) architectures have been quite effective and popular. We emphasize that these are powerful networks with applications in phenomics.

Software Platforms for Deep Learning

One of primary reasons for the resounding success of DL in an extremely short period of time is due to a relatively short start (from conception of new ideas to public dissemination of the software) to finish (diverse applications of these concepts) pipeline. This is built on the ML community's conscious choice of disseminating the software implementations of evolving DL models using open source software platforms specifically built for efficient implementation. This democratization involves the usage of hardware such as NVIDIA GPUs (such as Titan X, Titan V, and P40) efficiently, which allows the tasks to be massively parallelized [i.e., general purpose computing using GPU (GPGPU)]. While we do not attempt to provide a comprehensive review of all the DL software platforms, a few key platforms are introduced below to help the plant science community to get started and to navigate the fast-evolving space of practicing DL.

One of the first software platforms for DL is called Theano [42,43]. Theano, which provided a good modular coding architecture, was widely used by the community right from its inception. While Theano initially focused on DBNs, DNNs (with fully connected layers), and deep autoencoders, Caffe [44] was introduced to efficiently implement DCNNs. After that, Google released its user-friendly, graphical approach called Tensorflow [45] to design and train deep models. Currently, Tensorflow remains one of the most popular DL software platforms in both the core algorithm development and practitioners' communities. However, the building process of DL models was further abstracted and simplified by the Keras platform [46], which basically acts as a wrapper supporting both Theano and Tensorflow backends. Keras became instantly popular

among the uninitiated and practitioners alike due to its simplicity and compactness. As Keras became more and more popular, additional backends were added, such as CNTK [47] developed by Microsoft and MXNet [48]. Therefore, Keras can be a good starting point for the plant science researchers aiming to leverage the DL advancements. However, significant modifications to deep models or the learning process may be a little challenging using Keras.

Until 2016, most of the packages used precompiled computational graphs for performing training and computations. Recently, slightly more advanced DL platforms (that allow dynamic compiling of the graph using Autograd algorithms derived using differential geometry) allow faster computations to be performed. Among these, PyTorch [49] and Chainer [50] are quite popular for efficient DL implementation. These tools can be useful for advanced development of methods specific to the plant science community.

Deep Learning in Plant Phenotyping

DL has resulted in a paradigm shift in image-based plant phenotyping. A wide range of DL architectures have been used in plant phenotyping, including DCNN [51], RCNN [52], GoogLeNet [52,53], ResNet [52], SegNet [54], AlexNet [52,53], VGGNet [52], ResNeXt [52], SqueezeNet [55], GAN [56], LeNet18 [52] and ZFNet [52]. DL architectures have performed well on a broad range of plant phenotyping tasks, such as plant identification based on leaf vein patterns [57], leaf counting [58], tassel counting in maize [59], plant stalk count and stalk width [60], panicle segmentation [61], root and shoot localization and feature detection [62], bloom detection [63], and plant recognition [64].

Deep Learning in Plant Stress Phenotyping

In this section, we comprehensively review papers using DL for plant stress phenotyping (Table 1). Using PlantVillage, an open data resource (https://plantvillage.psu.edu/), Hughes and Salathé [65] and Mohanty *et al.* [53] utilized 54 306 images to train DCNN (AlexNet and GoogLeNet architectures) to identify 26 diseases from 14 distinct crop species. The mean F1 score was used as an assessment metric, with GoogLeNet outperforming AlexNet. The trained model using AlexNet (training from scratch) reached an accuracy of 85.5%, whereas the trained model using colored images in GoogLeNet (transfer learning) achieved an accuracy of 99.35%. Working with the same PlantVillage dataset, Ferentinos [66] used the VGG CNN and achieved a success rate of 99.5% in identifying plant stress. Results indicated that the model performance was better when original images were used. However, the use of preprocessed images did not cause noticeable reduction in prediction accuracy but led to significantly reduced computational time. Models were less robust when images from field and greenhouse were used to cross-validate, indicating the importance of model development on the target application scenario. The authors emphasize the need to maximize the number of images representing real conditions in the training data for generating models useful for field scouting. Both authors [53,66] provide evidence for eventual smartphone-based deployment for disease scouting, as the learnt model can be rapidly queried (less than a second on a CPU). However, the major limitations are the use of leaves compared to canopies for datasets and the need for an even more diverse species and stress image dataset.

With an aim to automate disease identification and classification, Amara *et al.* [67] deployed an LeNet architecture on image datasets. They chose three categories – healthy, black sigatoka, and black speckle (biotic stresses) – and used open source data from PlantVillage. DL was effective for identification and classification, even under varying and challenging conditions of diverse images: illumination, complex background, resolution, size, and orientation. They

showed good performance of their model for disease identification and two-class classification (healthy vs. diseased) and used a small learning rate for more precise results.

In other applications, the AlexNet model was used for multiple disease identification for apple leaf diseases (mosaic, rust, brown spot, and *Alternaria* leaf spot) and achieved an overall accuracy of 97.6% for disease identification [68]. In another study, an Inception-v3 model was trained on five cassava diseases and showed promising accuracy to identify three cassava diseases (cassava brown streak, cassava mosaic, and brown leaf spot) and two mite classes (green and red mite damage) in the field with an overall accuracy of 93% on the test dataset [69].

In order to identify various tomato diseases, a novel approach of combining super-resolution with conventional images was used to enhance the spatial resolution of diseased images. A super-resolution convolutional neural network (SRCNN) was used for super-resolution and it outpaced other conventional disease classification methods [70]. Another novel DL and data fusion approach for the automatic identification of olive 'quick decline' syndrome was designed using a DNN model called 'abstraction-level fusion' [71] and achieved a rate of detection of over 98%. To identify various viral diseases in cucumber, a CNN-based classifier was designed using images of seven viral diseases (melon yellow spot virus, zucchini yellow mosaic virus, cucurbit chlorotic yellows virus, cucumber mosaic virus, papaya ring spot virus, watermelon mosaic virus, and green mottle mosaic virus). This model achieved 82.3% accuracy with fourfold cross-validation [72]. A deep CNN model was trained using Caffe to identify 13 different plant diseases. The trained model accomplished precision in the range of 91–98% for separate class tests, with an average accuracy of 96.3%. A new architecture – called deep multiple instance learning (DMIL) – was deployed on the Wheat Disease Database 2017 (WDD2017) which consists of 9230 images across seven different classes (six common wheat diseases and healthy wheat). Here the VGG-FCN outperforms VGG-CNN architectures on identification accuracy for disease classes [73]. A DL architecture was trained to identify *Fusarium* wilt of radish using unmanned aerial vehicle (UAV) images. The model was able to detect *Fusarium* wilt in radish with high accuracy [74]. In another study, UAV images were used to train a CNN model to detect northern leaf blight (NLB) of maize in field conditions. The trained model could detect NLB with high accuracy (96.7%) [75]. Such trained models deployed on UAVs and unmanned ground vehicles (UGVs) show great promise for automated high-throughput phenotyping (HTP), precision breeding for resistant genotypes and for reducing the use of pesticide by allowing targeted application of pesticide for a sustainable environment.

A DL approach was used in tomato for detection of biotic stresses (diseases like gray mold, leaf mold, canker, plague, powdery mildew, and pests such as leaf miner and whitefly) and abiotic stresses (such as nutritional excess and low temperature). In order to develop a real-time system capable of identifying tomato stresses in field settings, a nondestructive imaging protocol was used to capture images under different illuminations, background, color, size, and shape of tomato fruit. Recent deep architectures from the ImageNet Challenge were used in this study, such as AlexNet, ZFNet, VGG-16, GoogLeNet, ResNet-50, ResNet-101, and ResNetXt-101, along with DL-based object detectors like Faster RCNN, R-FCN, and SSD. Faster RCNN with VGG-16 performed better in comparison to other architectures in identifying stresses like leaf mold, grey mold canker, leaf miner, and white fly [52].

To identify and visualize biotic and abiotic stresses in tomato, images of various viral (yellow leaf curl virus and tomato mosaic virus), bacterial (bacterial spot), and fungal (target spot, leaf mold, late blight, and early blight) diseases and pests like spider mites were obtained from the public PlantVillage database. A total of 14 828 images across nine tomato stresses were used to train

deep architectures like AlexNet and GoogLeNet. Following this pretraining step on the ImageNet dataset [76] to initialize the network weights, fine tuning was done by replacing the output layer with the nine stresses. Here, GoogLeNet outperformed AlexNet in terms of accuracy [77].

In order to automatically diagnose the fine-grained plant disease severity of the apple black rot, Wang *et al.* [78] trained a DL model using severity labels to annotate the healthy and black rot images of apple leaf obtained from the public PlantVillage dataset. In their study, they used a small training set of images for four stages of apple black rot, namely healthy (110 images), early- (137 images), middle- (180 images), and late-stage disease (125 images). They compared the performances of shallow networks trained from scratch with deep models VGGNet [79], Inception-v3 [26], and ResNet50 [80] fine-tuned through transfer learning (see later section). Out of these four models, the fine-tuned VGG-16 model performs the best, achieving an accuracy of 90.4% on test data, indicating the usefulness of the DL model even with a smaller training dataset in the automatic estimation of plant disease severity.

In a recent study, a DCNN framework was developed to identify, classify, and quantify eight soybean stresses using an explainable deep learning framework [81]. This novel framework uses an unsupervised approach to accurately isolate the visual cues representing the stress regions on the plant leaf. These visual cues are the regions that the DCNN model maximally uses for decision-making. Unsupervised localization of visual cues is used to quantify the stress. To the best of our knowledge, this is a first-of-its-kind plant stress quantification scheme, which avoids detailed and expensive stress region annotations by domain experts and opens up applications in image-based phenotyping, phenomics, and precision agriculture.

A supervised ML approach using hand-crafted feature extraction was used for automated severity classification (into five stress classes) and quantification (on a scale of 1–100%) for abiotic stress (iron deficiency chlorosis in soybean) using a smartphone app–based framework. A hierarchical classification with multiclass SVM was utilized [82,83]. We note that, while these strategies are useful, DL-based frameworks automate the feature extraction and classification step, providing a great advantage for automated plant stress phenotyping. This is especially important for HTP tools like UAV and UGV for real-time plant stress phenotyping.

We emphasize that literature is scarce in the prediction category of the ICQP paradigm. Prediction requires a combination of learning the present patterns to predict the future evolution of stresses. In the case of plant stress, this implies determining the presence of disease even when there are no apparent visual cues at the time of assessment. This is probably the most difficult part of plant stress phenotyping. In a recent study, Kaneda *et al.* [84] used a multimodal sliding window–based SVM (SW-SVM) for prediction of water stress using environmental and plant image data. The image feature is called ROAF (remarkable moving objects detected by adjacent optical flow) that detects moving objects through adjacent optical flow in changes in leaf dynamics and enables DNNs to extract the plant status of wilted leaves with water stress based on plant wilting motion, leading to precise and stable water stress prediction. The multimodal SW-SVR method uses two kinds of data: image data extracted using DNNs, in which water stress variation is expressed directly as plant wilting (and captured through ROAF), and environmental data related to plant transpiration, which is one of the causes of plant wilting.

Transfer Learning in Deep Learning

Transfer learning is an ML technique where the learnt model (and associated representations) from one problem is transferred to a dissimilar but allied problem. For example, knowledge

gained while learning to recognize soybean diseases can be used to recognize non-soybean species diseases, which happens due to the conserved nature of features across diseases making the transfer agnostic (i.e., generalization) to its implementation on different plant species. The nontrivial nature of data collection and sensing and the enormous computing resources required for training DL models makes transfer learning most desirable. This makes the utility and inference scope from learned models much wider than an isolated model specific to individual plant species. Transfer learning models are usually generated for a specific task, such as a computer vision in plant phenotyping problem [82], and can subsequently be used for training similar models for related problems. Transfer learning also enables rapid progress and improved performance in modeling subsequent tasks. The performance training relationship can be used to determine the benefits of transfer learning and are elucidated with higher start (higher performance, i.e., initial skill of source model), steeper slope (more rapid gain in performance), and higher asymptote (highest performance with compared to without learning) [85]; the best transfer learning happens when all three are optimal.

There are three main approaches to transfer learning [51,72,75,81,86]. In the context of DL transfer learning, this may involve (i) reusing the entire model (architecture + parameters) for the new task that can involve a new dataset, (ii) only reusing the model architecture/hyperparameters, but learning the parameters from scratch with the new dataset, and (iii) reusing a model but after fine tuning the model parameters based on the new dataset – this process is often referred to as domain adaptation.

The type of transfer learning to use on new dataset depends on two factors: whether the new dataset is small or big; and how similar it is to the data used to train the original model. For example, the DeepFace [87] pretrained model was generated on human face classification and may not be the best pretrained model for plant disease images. This inconsistency happens due to markedly different image content and features. An additional factor to be vigilant about is overfitting. However, similarity of training data to original data leads to easier transferability. In an ideal situation of large datasets in both original and experimental sets, even dissimilarity between the datasets is less of an issue due to the ability to develop a model from scratch. However, for all practical purposes, it may still be advisable to initialize weights from a pretrained model, as it allows more robust fine-tuning throughout the entire network. In this regard, smaller learning rates for convolutional nets are common as network parameters should not be changed dramatically.

Concluding Remarks and Future Perspectives

ML (and more specifically DL) approaches show great promise for improving the speed, accuracy, reliability, and scalability of disease phenotyping to accomplish diverse programmatic goals. In this review, we focus on DL approaches for image data as this is an area where transformative impact in plant science is possible (due to the confluence of available cheap digital imaging, Internet of Things capabilities, and computing and data storage capabilities). End-to-end DL approaches can greatly streamline the full spectrum of image-based plant stress phenotyping from identification and classification to even quantification [81,86]. DL approaches are well positioned to be integrated with other imaging modalities. Plant phenotyping using hyperspectral imaging is a particularly promising avenue, where the individual datasets (i.e., each hyperspectral cube) themselves become quite large. Here novel DL approaches, for example 3D CNN architectures, would be promising candidates. These imaging modalities when fused with digital image data could potentially allow early detection of plant stress, before the appearance of visible symptoms (or rather human interpretable visible symptoms). We finally note that the rapid progress in, and availability of, DL workflows now

Outstanding Questions

While DL has been used for identification, classification, and, more recently, quantification, it has not been used for prediction. Can DL advances be leveraged to predict future plant stresses?

DL under annotation cost constraints: it appears that the major bottleneck for DL is the lack of availability of large training data. Can we develop smart strategies to reduce the need for expensive labeling?

DL for heterogeneous information fusion: increasing availability of multiple types, scales, and resolution of data. Can we leverage DL to fuse heterogeneous datasets for plant phenotyping?

Efficient deployment: how can we deploy trained DL models under typical data storage constraints in rural landscapes?

Box 1. Best Practices for Using Deep Learning in Plant Stress Phenotyping

Data

- A sufficient amount of balanced data is required.
- If the above is not available, boot strapping and augmentation needs to be considered.
- Annotation, augmentation, outlier rejection, standardization, and normalization of data are strongly encouraged.

Architecture

- The problem needs to be clearly defined, which decides the input and output.
- Proper architecture should be chosen based on the size and type of input data and the complexity of the problem.
- Computational hardware availability considerations also determine network choices.
- Computation resource availability and allowable wait time for solutions.

Training Process

- Exploration of hyperparameters including optimization protocol, network architecture, batch size, objective function, learning rate, and regularization schemes.
- Consider using transfer learning.

Result Evaluation

- Performance evaluation with sufficient validation data to ensure low overfitting.
- Check for robustness under data and model perturbation.
- Intermediate feature visualization should be performed to build confidence about the model.

makes it feasible to develop and deploy application-specific models. Thus, high-throughput phenotyping in the field can be as accessible as phenotyping in controlled laboratory settings with the collection and curation of field-based phenotyping datasets.

We have identified some best practices and practitioner tips to successfully deploy DL concepts for plant science applications (Box 1). Here, we encourage the creation and broad availability of community-wide curated, labeled datasets which would rapidly accelerate deployment (as well as enhance interactions with ML experts who would use these curated datasets as benchmark data, similar to MNIST, ImageNet, or CIFAR). The creation of such open, labelled, broad-spectrum plant stress datasets across different plant species will avoid duplication in data collection, enabling plant scientists with smaller datasets to utilize DL by using transfer learning. We also strongly encourage using imaging data from field conditions (i.e., with changes in illumination, shading, and occlusions) for creating training datasets, as this will create robust models that can be embedded onto high-throughput systems (like UAVs and other autonomous systems). The confluence of imaging platforms advances in DL approaches and the availability of storage and computing resources makes this an exciting time to be pursuing plant phenotyping including stress-related traits. We conclude by identifying a few critical outstanding problems for the community to consider (see Outstanding Questions).

Acknowledgments

This work was supported by the Iowa State University (Presidential Initiative for Interdisciplinary Research) to all authors; the Plant Science Institute to A.K.S., B.G., S.S.; the Iowa Soybean Association (A.S., A.K.S.); Monsanto Chair in Soybean Breeding (A.K.S.); R F Baker Center for Plant Breeding (A.K.S.), United States Department of Agriculture – NIFA project to all authors; and USDA-CRIS IOW04314 project (A.K.S., A.S.). We thank Sambuddha Ghosal for assisting with illustrations and Aditya Balu and Adedotun Akintayo for inputs on deep learning architectures.

Supplemental Information

Supplemental information associated with this article can be found online at https://doi.org/10.1016/j.tplants.2018.07.004.

References

1. Singh, A. *et al.* (2016) Machine learning for high-throughput stress phenotyping in plants. *Trends Plant Sci.* 21, 110–124
2. Bock, C. *et al.* (2010) Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging. *Crit. Rev. Plant Sci.* 29, 59–107
3. Madden, L.V. *et al.* (2007) *The Study of Plant Disease Epidemics,* American Phytopathological Society
4. Nutter, F.W. *et al.* (1993) Assessing the accuracy, intra-rater repeatability, and inter-rater reliability of disease assessment systems. *Phytopathology* 83, 806–812
5. Pethybridge, S.J. and Nelson, S.C. (2017) Estimate, a new iPad application for assessment of plant disease severity using photographic standard area diagrams. *Plant Dis.* 102, 276–281
6. Hallau, L. *et al.* (2018) Automated identification of sugar beet diseases using smartphones. *Plant Pathol.* 67, 399–410
7. Pethybridge, S.J. and Nelson, S.C. (2015) Leaf Doctor: a new portable application for quantifying plant disease severity. *Plant Dis.* 99, 1310–1316
8. Goodfellow, I. *et al.* (2016) *Deep Learning,* MIT Press
9. Kingma, D.P. and Ba, J. (2014) Adam: A method for stochastic optimization. *CoRR* abs/1412.6980
10. Taylor, G., *et al.* (2016) Training neural networks without gradients: a scalable ADMM approach. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning* (Vol. 48), pp. 2722–2731, JMLR
11. Schmidhuber, J. (2015) Deep learning in neural networks: an overview. *Neural Netw.* 61, 85–117
12. Fukushima, K. (1979) Neural-network model for a mechanism of pattern recognition unaffected by shift in position. *Trans. IECE Japan* J62-A, 658–665
13. Lecun, Y. *et al.* (1998) Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 2278–2324
14. Hinton, G.E. and Salakhutdinov, R.R. (2006) Reducing the dimensionality of data with neural networks. *Science* 313, 504–507
15. Cireşan, D., *et al.* (2012) Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642-3649, IEEE
16. Cireşan, D.C. *et al.* (2010) Deep, big, simple neural nets for handwritten digit recognition. *Neural Comput.* 22, 3207–3220
17. Krizhevsky, A., *et al.* (2012) ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems, NIPS 2012* (Vol. 1) (Pereira, F. *et al.*, eds), pp. 1097–1105, Curran Associates Inc.
18. Zeiler, M.D. and Fergus, R. (2013) Visualizing and understanding convolutional networks. *CoRR* abs/1311.2901
19. Simonyan, K. and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556
20. Szegedy, C. *et al.* (2016) Inception-v4, Inception-ResNet and the impact of residual connections on learning. *CoRR* abs/1602.07261
21. Chollet, F. (2016) Xception: deep learning with depthwise separable convolutions. *CoRR* abs/1610.02357
22. He, K. *et al.* (2015) Deep residual learning for image recognition. *CoRR* abs/1512.03385
23. Srivastava, N. *et al.* (2014) Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1929–1958
24. Ioffe, S. and Szegedy, C. (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. *arXiv* 1502.03167
25. Chollet, F. (2017) *Deep Learning with Python,* Manning Publications Company
26. Szegedy, C. *et al.* (2015) Rethinking the Inception architecture for computer vision. *CoRR* abs/1512.00567
27. Girshick, R., *et al.* (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, IEEE
28. Girshick, R.B. (2015) Fast R-CNN. *CoRR* abs/1504.08083
29. Girshick, R.B. *et al.* (2013) Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR* abs/1311.2524
30. Ren, S. *et al.* (2015) Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR* abs/1506.01497
31. He, K. *et al.* (2017) Mask R-CNN. *CoRR* abs/1703.06870
32. Shelhamer, E. *et al.* (2017) Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 640–651
33. Liu, W. *et al.* (2015) SSD: single shot multibox detector. *CoRR* abs/1512.02325
34. Russakovsky, O. *et al.* (2015) ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* 115, 211–252
35. Redmon, J. and Farhadi, A. (2016) YOLO9000: better, faster, stronger. *CoRR* abs/1612.08242
36. Mohamed, A. *et al.* (2012) Acoustic modeling using deep belief networks. *IEEE Trans. Audio Speech Lang. Process.* 20, 14–22
37. Larochelle, H. *et al.* (2012) Learning algorithms for the classification restricted Boltzmann machine. *J. Mach. Learn. Res.* 13, 643–669
38. Akintayo, A. *et al.* (2018) A deep learning framework to discern and count microscopic nematode eggs. *Sci. Rep.* 8, 9145
39. Lore, K.G. *et al.* (2015) LLNet: a deep autoencoder approach to natural low-light image enhancement. *CoRR* abs/1511.03995
40. Pascanu, R. *et al.* (2013) How to construct deep recurrent neural networks. *CoRR* abs/1312.6026
41. Graves, A. *et al.* (2013) Speech recognition with deep recurrent neural networks. *CoRR* abs/1303.5778
42. Bergstra, J., *et al.* (2010) Theano: a CPU and GPU math compiler in Python. In *Proceedings of the 9th Python in Science Conference* (van der Walt, S. and Millman, J., eds), pp. 3–10
43. Bastien, F. *et al.* (2012) Theano: new features and speed improvements. *arXiv* 1211.5590
44. Jia, Y., *et al.* (2014) Caffe: convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM International Conference on Multimedia*, pp. 675–678, ACM
45. Mart, *et al.* (2016) TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, pp. 265–283, USENIX Association
46. Chollet, F. (2015) Keras. *GitHub* Published online 21 November, 2015. https://github.com/fchollet/keras
47. Seide, F., and Agarwal, A. (2016) CNTK: Microsoft's open-source deep learning toolkit. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2135–2135, ACM
48. Chen, T. *et al.* (2015) Mxnet: a flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv* 1512.01274
49. Team, P.C. (2017) Pytorch: tensors and dynamic neural networks in Python with strong GPU acceleration. *GitHub* pub online: March 22, 2017. https://github.com/pytorch/pytorch
50. Tokui, S., *et al.* (2015) Chainer: a next-generation open source framework for deep learning. In *Proceedings of the Workshop on Machine Learning Systems (LearningSys) at the 28th Annual Conference on Neural Information Processing Systems (NIPS)*. http://learningsys.org/papers/LearningSys_2015_paper_33.pdf
51. Pound, M.P. *et al.* (2017) Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *GigaScience* 6, 1–10
52. Fuentes, A. *et al.* (2017) A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors (Basel, Switzerland)* 17, 2022

53. Mohanty, S.P. *et al.* (2016) Using deep learning for image-based plant disease detection. *Front. Plant Sci.* 7, 1419 http://dx.doi.org/10.3389/fpls.2016.01419
54. Aich, S. *et al.* (2017) Leaf counting with deep convolutional and deconvolutional networks. *CoRR* abs/1708.07570
55. Durmuş, H., *et al.* (2017) Disease detection on the leaves of the tomato plants by using deep learning. In *2017 6th International Conference on Agro-Geoinformatics*, pp. 1–5, IEEE
56. Giuffrida, M.V. *et al.* (2017) ARIGAN: synthetic *Arabidopsis* plants using generative adversarial network. *CoRR* abs/1709.00938
57. Grinblat, G.L. *et al.* (2016) Deep learning for plant identification using vein morphological patterns. *Comput. Electron. Agric.* 127, 418–424
58. Ubbens, J. *et al.* (2018) The use of plant models in deep learning: an application to leaf counting in rosette plants. *Plant Methods* 14, 6
59. Lu, H. *et al.* (2017) TasselNet: counting maize tassels in the wild via local counts regression network. *Plant Methods* 13, 79
60. Baweja, H., *et al.* (2018) StalkNet: a deep learning pipeline for high-throughput measurement of plant stalk count and stalk width. In *Field and Service Robotics*. *Springer Proceedings in Advanced Robotics* (Vol. 5) (Hutter, M. and Siegwart, R. eds), pp. 271–284, Springer
61. Xiong, X. *et al.* (2017) Panicle-SEG: a robust image segmentation method for rice panicles in the field based on deep learning and superpixel optimization. *Plant Methods* 13, 104
62. Pound, M.P. *et al.* (2016) Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *Gigascience* 6, 1–10
63. Xu, R. *et al.* (2018) Aerial images and convolutional neural network for cotton bloom detection. *Front. Plant Sci.* 8, 2235
64. Šulc, M. and Matas, J. (2017) Fine-grained recognition of plants from images. *Plant Methods* 13, 115
65. Hughes, D.P. and Salathé, M. (2015) An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing. *CoRR* abs/1511.08060
66. Ferentinos, K.P. (2018) Deep learning models for plant disease detection and diagnosis. *Comput. Electron. Agric.* 145, 311–318
67. Amara, J. *et al.* (2017) A deep learning-based approach for banana leaf diseases classification. In *Lecture Notes in Informatics (LNI)*, pp. 79–88, Gesellschaft für Informatik
68. Liu, B. *et al.* (2018) Identification of apple leaf diseases based on deep convolutional neural networks. *Symmetry* 10, 11
69. Ramcharan, A. *et al.* (2017) Deep learning for image-based cassava disease detection. *Front. Plant Sci.* 8, 1852
70. Yamamoto, K. *et al.* (2017) Super-resolution of plant disease images for the acceleration of image-based phenotyping and vigor diagnosis in agriculture. *Sensors (Basel)* 17, E2557
71. Cruz, A.C. *et al.* (2017) X-FIDO: an effective application for detecting olive quick decline syndrome with deep learning and data fusion. *Front. Plant Sci.* 8, 1741
72. Fujita, E., *et al.* (2016) Basic investigation on a robust and practical plant diagnostic system. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 989–992, IEEE
73. Lu, J. *et al.* (2017) An in-field automatic wheat disease diagnosis system. *CoRR* abs/1710.08299
74. Ha, J.G. *et al.* (2017) Deep convolutional neural network for classifying *Fusarium* wilt of radish from unmanned aerial vehicles. *J. Appl. Remote Sens.* 11, 042621
75. DeChant, C. *et al.* (2017) Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning. *Phytopathology* 107, 1426–1432
76. Deng, J., *et al.* (2009) ImageNet: a large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, IEEE
77. Brahimi, M. *et al.* (2017) Deep learning for tomato diseases: classification and symptoms visualization. *Appl. Artif. Intell.* 31, 299–315
78. Wang, G. *et al.* (2017) Automatic image-based plant disease severity estimation using deep learning. *Comput. Intell. Neurosci.* 2017, 2917536
79. Simonyan, K. and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556
80. He, K. *et al.* (2015) Deep residual learning for image recognition. *arXiv* 1512.03385
81. Ghosal, S. *et al.* (2018) An explainable deep machine vision framework for plant stress phenotyping. *Proc. Natl. Acad. Sci. U. S. A.* 115, 4613–4618
82. Zhang, J. *et al.* (2017) Computer vision and machine learning for robust phenotyping in genome-wide studies. *Sci. Rep.* 7, 44048
83. Naik, H.S. *et al.* (2017) A real-time phenotyping framework using machine learning for plant stress severity rating in soybean. *Plant Methods* 13, 23
84. Kaneda, Y. *et al.* (2017) Multi-modal sliding window-based support vector regression for predicting plant water stress. *Knowl. Based Syst.* 134, 135–148
85. Torrey, L. and Shavlik, J. *et al.* (2009) Transfer learning. In *Handbook of Research on Machine Learning Applications* (Soria, E., ed.), pp. 242–264, IGI Global
86. Xiong, X. *et al.* (2017) Panicle-SEG: a robust image segmentation method for rice panicles in the field based on deep learning and superpixel optimization. *Plant Methods* 13, 104
87. Taigman, Y., *et al.* (2014) DeepFace: closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, IEEE
88. Sladojevic, S. *et al.* (2016) Deep neural networks based recognition of plant diseases by leaf image classification. *Comput. Intell. Neurosci.* 2016, 3289801