

Seed-per-pod estimation for plant breeding using deep learning

STUDENTS: MELINA PRADO
GUSTAVO LIMA RODRIGUES

TEACHERS: PROF. DR. JOSÉ BALDIN PINHEIRO
PROF. DR. FERNANDO ANGELO PIOTO



Paper



Contents lists available at [ScienceDirect](#)

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag



Original papers

Seed-pod estimation for plant breeding using deep learning

L.C. Uzal^{a,*}, G.L. Grinblat^a, R. Namías^a, M.G. Larese^a, J.S. Bianchi^b, E.N. Morandi^b, P.M. Granitto^a

^a CIFASIS, French Argentine International Center for Information and Systems Sciences, CONICET - UNR, Argentina

^b Laboratorio de Fisiología Vegetal, Facultad de Ciencias Agrarias, Universidad Nacional de Rosario & Instituto de Investigaciones en Ciencias Agrarias de Rosario (IICAR), CONICET - UNR, Argentina

Fonte: Uzal *et. al.*, 2018.

Plant phenotyping



Fonte: Agriexpo, 2019.



Fonte: Embrapa, 2016.

Three major componentes of a soybean crop:

Pods per plant (PN)



Fonte: Technologytimes, 2014.

Seeds per pod (SPP)



Fonte: Sistema Faep, 2018.

Seed size (SS)

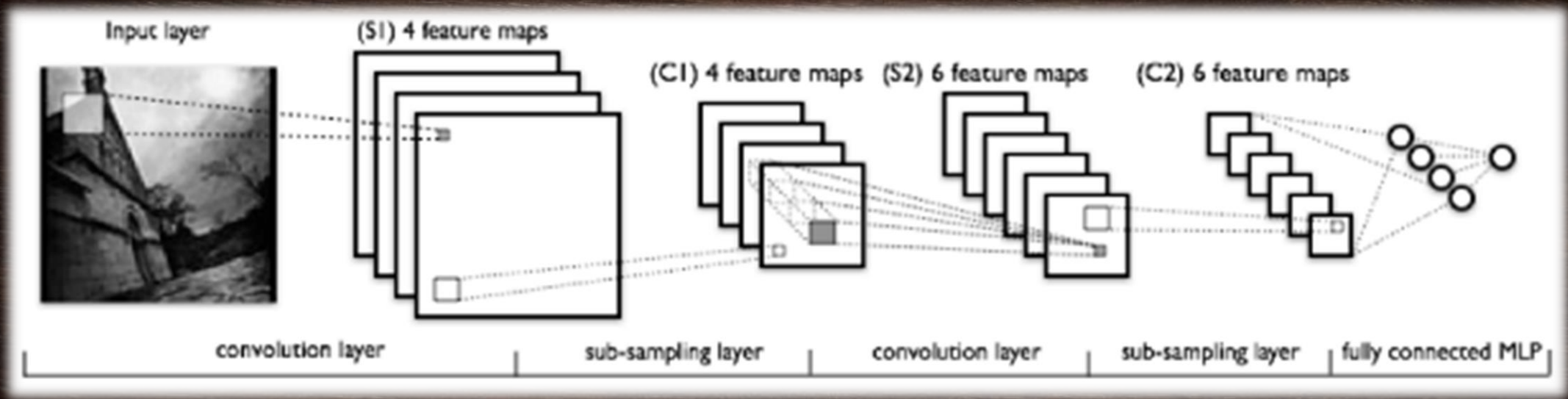


Fonte: Maissoja, 2018.

Classic procedure

- “Define and extract appropriate features for the problem at hand and then train a classifier” (UZAL, 2018)
- Three state-of-the-art classic classification methods were implemented:
 - SVM
 - Random Forest (RF)
 - Penalized Discriminant Analysis (PDA)

Convolutional neural networks



Fonte: Ravindra, S., 2017.

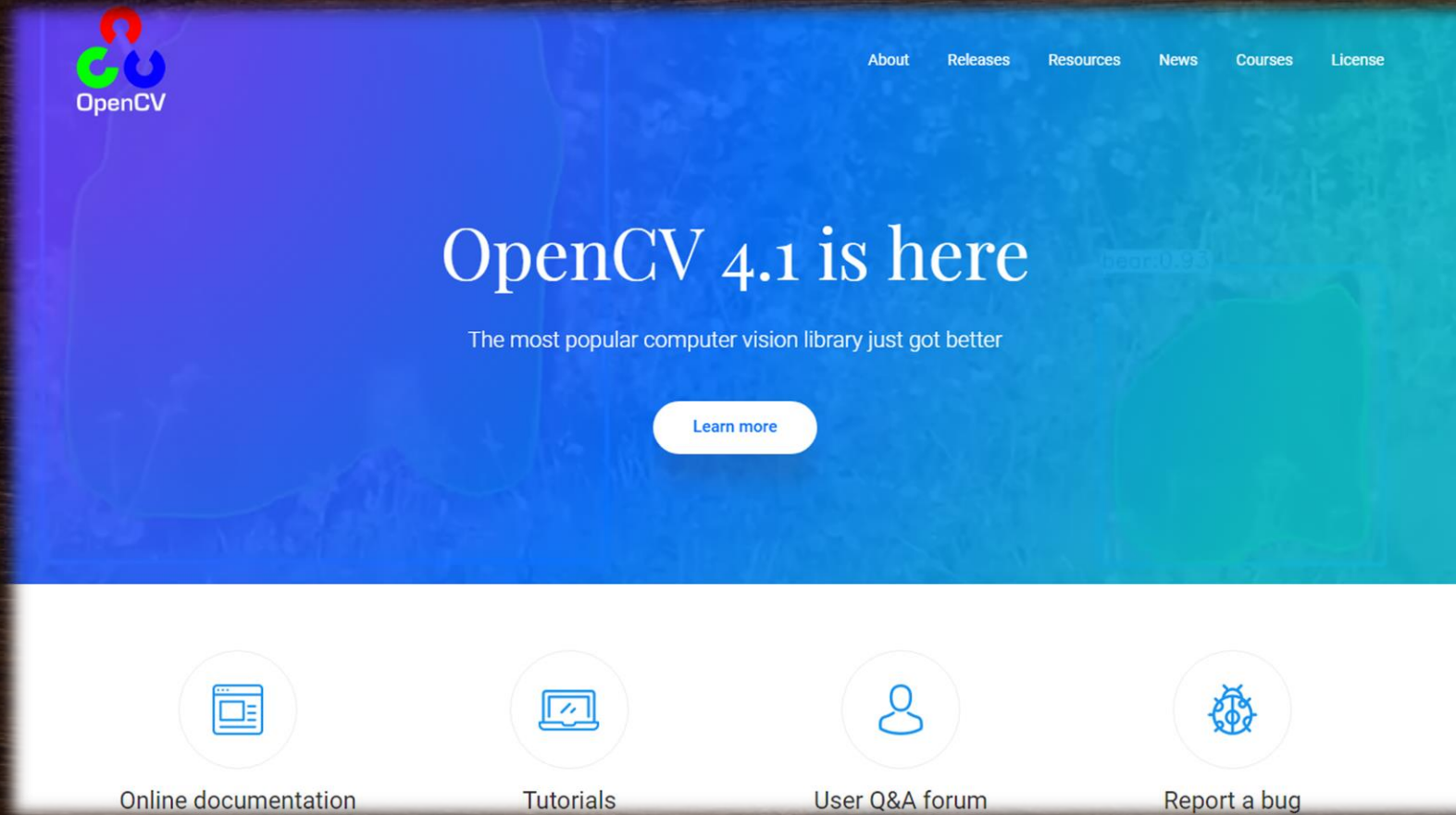
Data collection



Fig. 1. Sample photographs of soybean pods used to build the dataset. Each pod is manually classified by an expert and photographed within its class group defined by the SPP number.

Fonte: Uzal *et. al.*, 2018.

Pods segmentation



Fonte: OpenCV, 2019.

Pods segmentation



Fig. 1. Sample photographs of soybean pods used to build the dataset. Each pod is manually classified by an expert and photographed within its class group defined by the SPP number.

Fonte: Uzal *et. al.*, 2018.

Pods segmentation

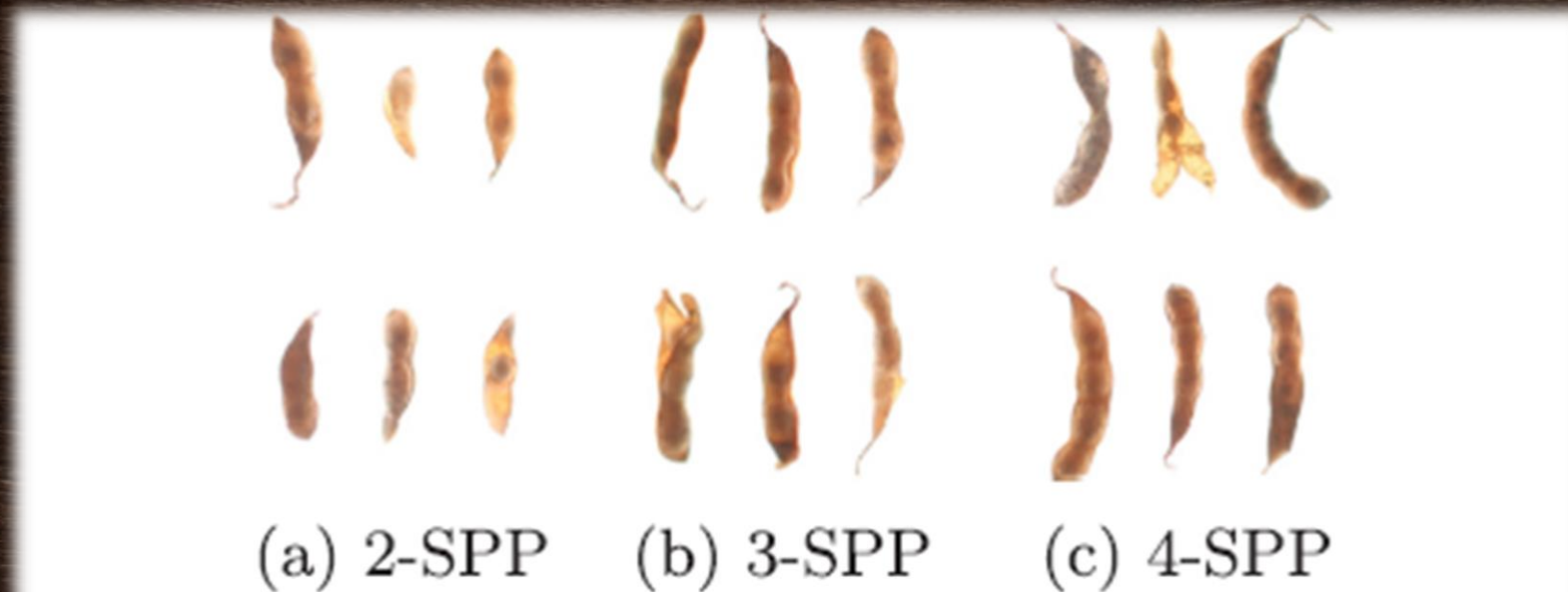


Fig. 2. Pod images obtained after segmentation process. Each panel corresponds to one of the three class label to be recognized, defined by the number of seeds per pod (SPP). Images shown preserve the original relative sizes.

Fonte: Uzal *et. al.*, 2018.

Pods segmentation

Table 1

Total number of examples corresponding to each class and season.

Class	Season 1	Season 2
2-SPP	811	3746
3-SPP	4598	5075
4-SPP	2444	1504
Total	7853	10325

Fonte: Uzal *et. al.*, 2018.

Preprocessing

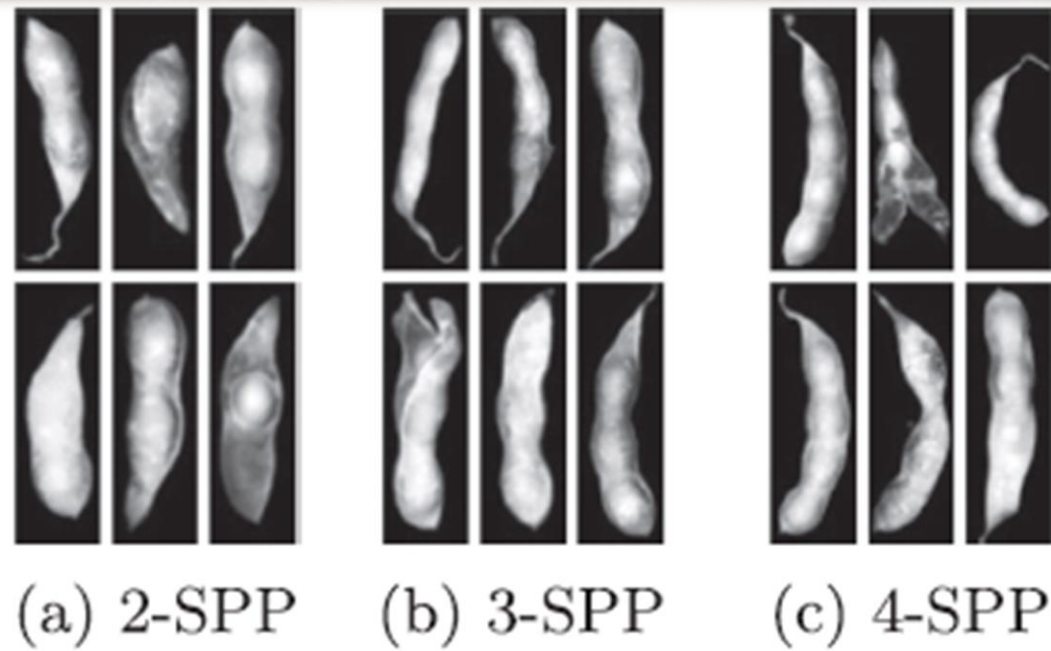
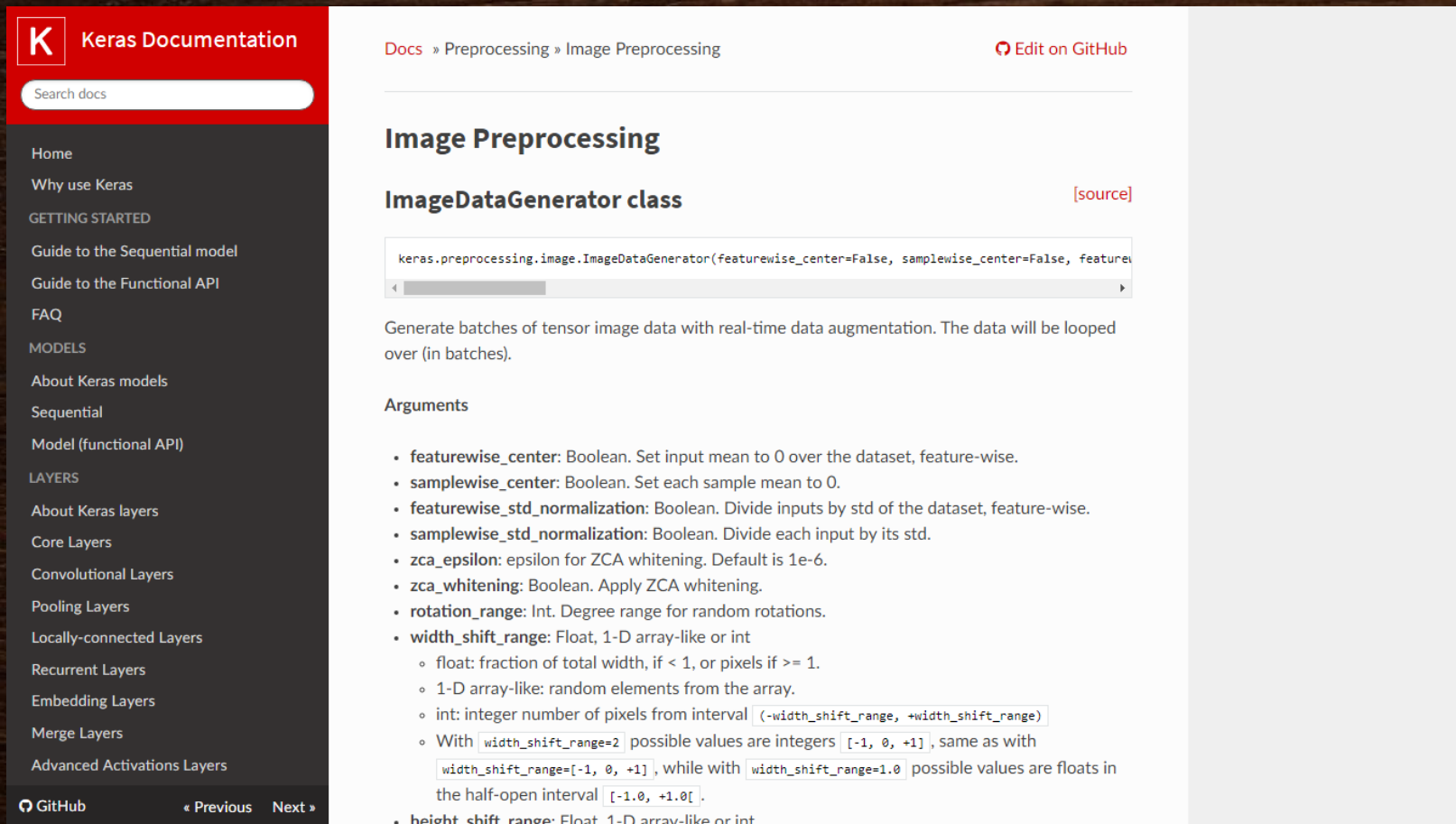


Fig. 3. Images obtained after preprocessing step. Samples are the same of Fig. 2.

Fonte: Uzal *et. al.*, 2018.

Data augmentation



The screenshot displays the Keras Documentation website. On the left is a dark sidebar with a red header containing the 'Keras Documentation' logo and a search bar. The sidebar lists various navigation options such as 'Home', 'Why use Keras', 'GETTING STARTED', 'MODELS', and 'LAYERS'. The main content area is white and shows the breadcrumb 'Docs » Preprocessing » Image Preprocessing' and an 'Edit on GitHub' link. The title 'Image Preprocessing' is followed by the 'ImageDataGenerator class' with a '[source]' link. A code block shows the class constructor: `keras.preprocessing.image.ImageDataGenerator(featurewise_center=False, samplewise_center=False, featurewise_std_normalization=False, samplewise_std_normalization=False, zca_epsilon=1e-6, zca_whitening=False, rotation_range=0, width_shift_range=0, height_shift_range=0, shear_range=0, zoom_range=0, horizontal_flip=False, vertical_flip=False, validation_split=0.1)`. Below the code, a paragraph explains that the class generates batches of tensor image data with real-time data augmentation. The 'Arguments' section lists several parameters:

- featurewise_center**: Boolean. Set input mean to 0 over the dataset, feature-wise.
- samplewise_center**: Boolean. Set each sample mean to 0.
- featurewise_std_normalization**: Boolean. Divide inputs by std of the dataset, feature-wise.
- samplewise_std_normalization**: Boolean. Divide each input by its std.
- zca_epsilon**: epsilon for ZCA whitening. Default is 1e-6.
- zca_whitening**: Boolean. Apply ZCA whitening.
- rotation_range**: Int. Degree range for random rotations.
- width_shift_range**: Float, 1-D array-like or int
 - float: fraction of total width, if < 1, or pixels if >= 1.
 - 1-D array-like: random elements from the array.
 - int: integer number of pixels from interval `(-width_shift_range, +width_shift_range)`
 - With `width_shift_range=2` possible values are integers `[-1, 0, +1]`, same as with `width_shift_range=[-1, 0, +1]`, while with `width_shift_range=1.0` possible values are floats in the half-open interval `[-1.0, +1.0[`.
- height_shift_range**: Float, 1-D array-like or int

Fonte: Keras, 2019.

Data augmentation



The screenshot shows the official website for GIMP (GNU Image Manipulation Program). The header features a dark green navigation bar with links for GIMP, DOWNLOAD, NEWS, ABOUT, DOCS, PARTICIPATE, TUTORIALS, and DONATE, along with social media icons for GitHub, YouTube, and Bitcoin. The main banner area has a blue background with the GIMP logo (a cartoon cat) and the text 'GIMP GNU IMAGE MANIPULATION PROGRAM'. Below the logo, there are two buttons: a red 'DOWNLOAD 2.10.10' button and a dark blue 'RELEASE NOTES' button. The content area is divided into two columns. The left column is titled 'The Free & Open Source Image Editor' and contains a paragraph describing GIMP as the official website of the GNU Image Manipulation Program, followed by a paragraph stating it is a cross-platform image editor available for GNU/Linux, OS X, Windows, and more, and is free software. The right column is titled 'Recent News' and lists three items: 'GIMP 2.10.10 Released' (2019-04-07), 'GIMP and GEGL in 2018' (2019-01-02), and 'GIMP 2.10.8 Released' (2018-11-08). A fourth item, 'GIMP receives a \$100K donation' (2018-08-30), is partially visible at the bottom of the list.

[GIMP](#) [DOWNLOAD](#) [NEWS](#) [ABOUT](#) [DOCS](#) [PARTICIPATE](#) [TUTORIALS](#) [DONATE](#) [P](#) [B](#)

 **GIMP** GNU IMAGE
MANIPULATION PROGRAM

[DOWNLOAD 2.10.10](#) [RELEASE NOTES](#)

The Free & Open Source Image Editor

This is the official website of the GNU Image Manipulation Program (GIMP).

GIMP is a cross-platform image editor available for GNU/Linux, OS X, Windows and more operating systems. It is free software, you can change its source code and distribute your changes.

Whether you are a graphic designer, photographer, illustrator, or scientist, GIMP provides you with sophisticated tools to get your job done. You can further enhance your productivity with GIMP thanks to many customization options and 3rd party plugins.

Recent News

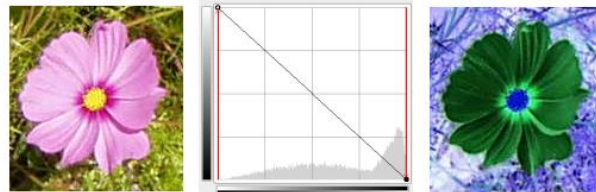
- [GIMP 2.10.10 Released](#)
2019-04-07
- [GIMP and GEGL in 2018](#)
2019-01-02
- [GIMP 2.10.8 Released](#)
2018-11-08
- [GIMP receives a \\$100K donation](#)
2018-08-30

Fonte: Gimp, 2019.

Data augmentation

8.11.3.2. Practical cases

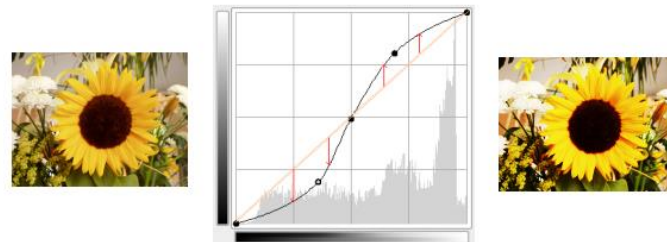
Invert colors



Inverted curve

Black is made White (fully colored / fully opaque). White is made black (black, fully transparent). All pixels adopt the complementary color. Why that? Because subtracting the channel values from 255 gives the complementary color. For example: 19;197;248 a sky blue gives 255-19; 255-197; 255-248 = 236;58;7, a bright red.

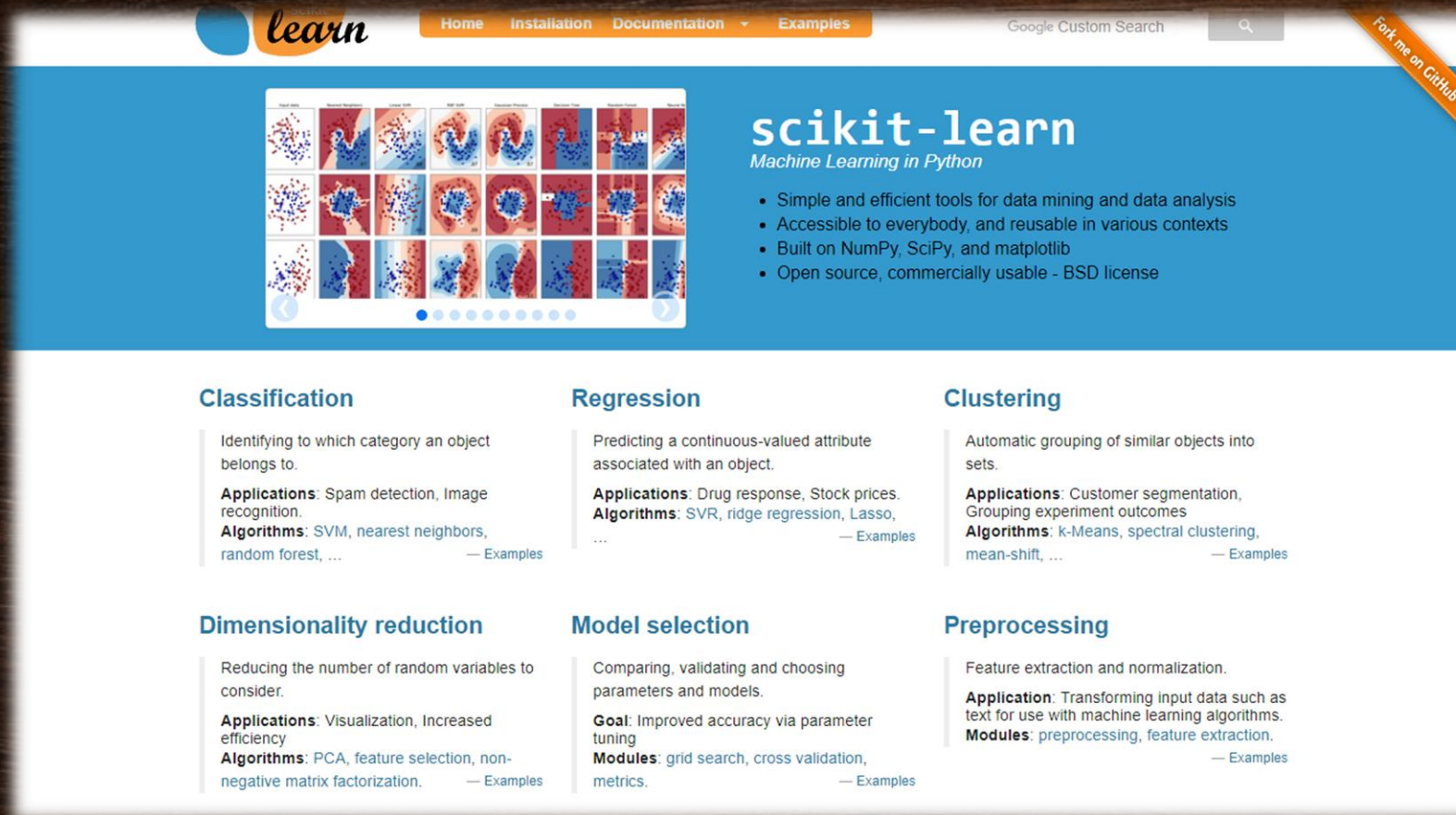
Enhance contrast



Feature extraction details

- Geometrical characteristics
 - Area
 - Perimeter
 - Major and minor axis length
- Shape features
 - Density
 - Elongation
 - Compactness
 - Rugosity
 - Axis ratio
- Etc

SVM implementation details



learn

Home Installation Documentation Examples

Google Custom Search

Fork me on GitHub

scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics. — Examples

Preprocessing

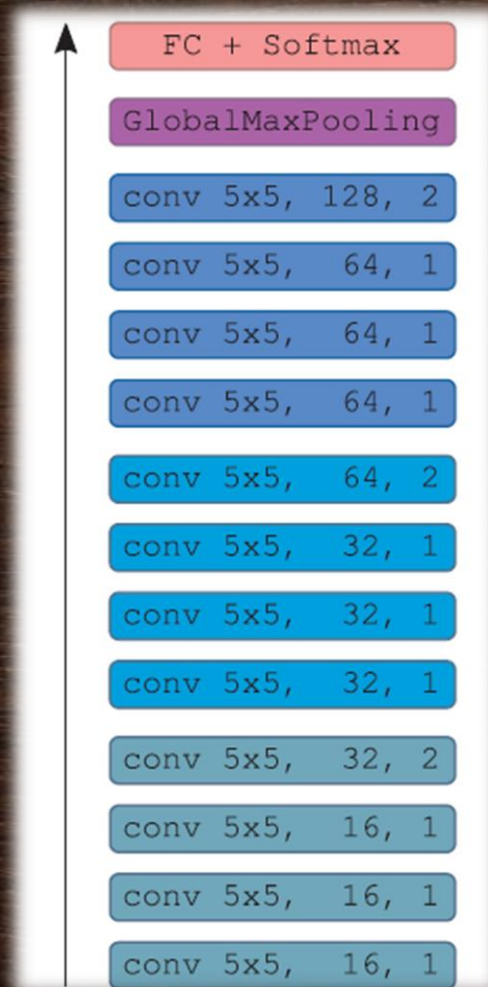
Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction. — Examples

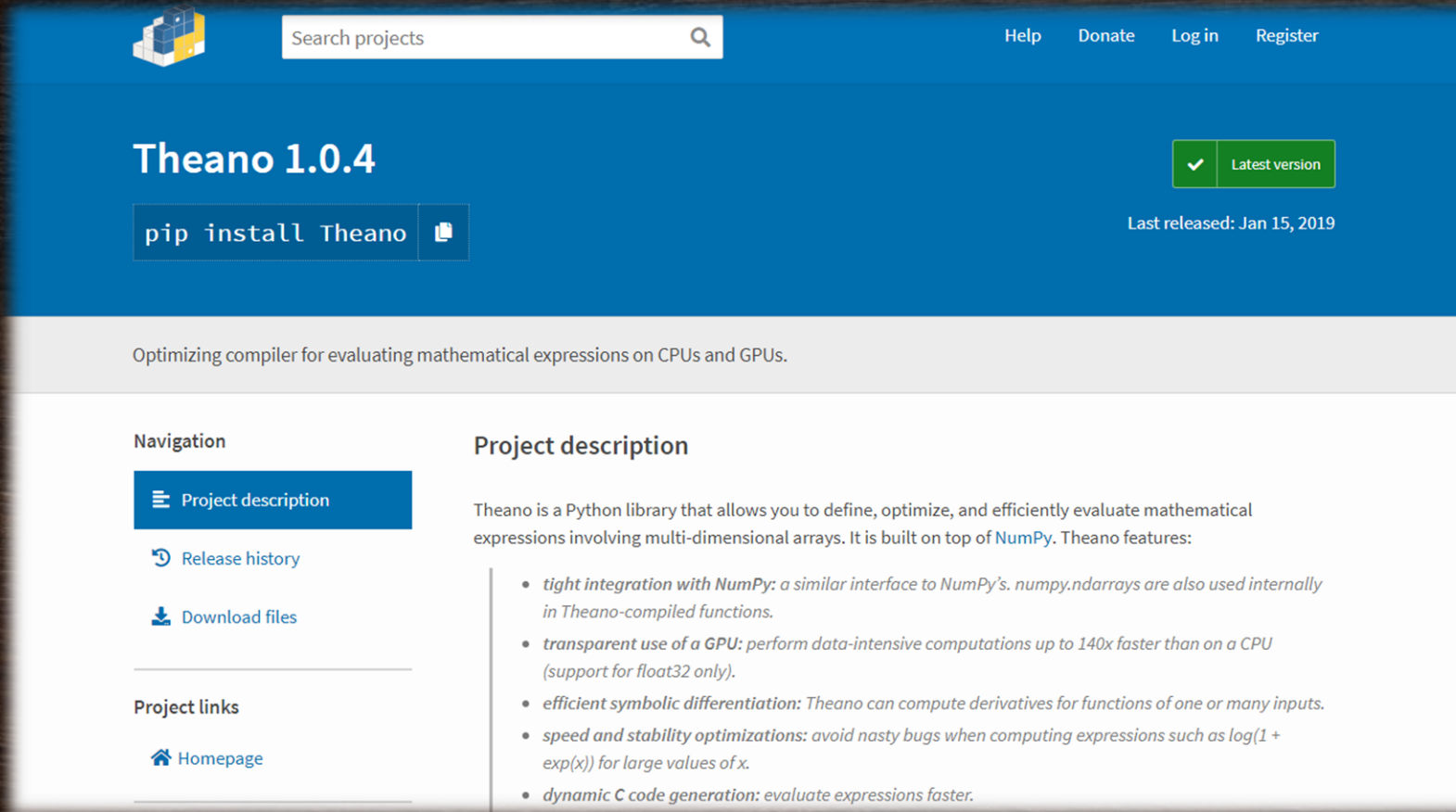
Fonte: Scikit-learn, 2019.

CNN implementation details




Fonte: Uzal *et. al.*, 2018.


CNN implementation details




The screenshot shows the PyPI page for Theano 1.0.4. The header is blue with the Theano logo, a search bar, and navigation links for Help, Donate, Log in, and Register. The main content area is white with a blue header for the project name 'Theano 1.0.4'. Below the name is a green button with a checkmark and the text 'Latest version', and a grey button with the text 'Last released: Jan 15, 2019'. A dark blue button contains the command 'pip install Theano' and a copy icon. The description below reads 'Optimizing compiler for evaluating mathematical expressions on CPUs and GPUs.' The left sidebar has a 'Navigation' section with 'Project description' (highlighted), 'Release history', and 'Download files'. Below that is a 'Project links' section with a 'Homepage' link. The main content area has a 'Project description' section with a paragraph and a bulleted list of features.

 Search projects [Help](#) [Donate](#) [Log in](#) [Register](#)

Theano 1.0.4

`pip install Theano` 

 Latest version

Last released: Jan 15, 2019

Optimizing compiler for evaluating mathematical expressions on CPUs and GPUs.

Navigation

- [Project description](#)
- [Release history](#)
- [Download files](#)

Project links

- [Homepage](#)

Project description

Theano is a Python library that allows you to define, optimize, and efficiently evaluate mathematical expressions involving multi-dimensional arrays. It is built on top of [NumPy](#). Theano features:

- *tight integration with NumPy*: a similar interface to NumPy's. `numpy.ndarrays` are also used internally in Theano-compiled functions.
- *transparent use of a GPU*: perform data-intensive computations up to 140x faster than on a CPU (support for float32 only).
- *efficient symbolic differentiation*: Theano can compute derivatives for functions of one or many inputs.
- *speed and stability optimizations*: avoid nasty bugs when computing expressions such as $\log(1 + \exp(x))$ for large values of x .
- *dynamic C code generation*: evaluate expressions faster.

Fonte: Theano, 2018.

CNN implementation details

The screenshot displays the Lasagne documentation website. The top navigation bar includes the 'Lasagne' logo, the version 'latest', and a search bar labeled 'Search docs'. A sidebar on the left lists various documentation sections such as 'Installation', 'Tutorial', 'Layers', and 'Development'. The main content area features a 'Welcome to Lasagne' heading, followed by a brief introduction and a 'User Guide' section with a bulleted list of links for installation, tutorial, and layers.

Lasagne

latest

Search docs

Installation

Tutorial

Layers

Creating custom layers

Development

lasagne.layers

lasagne.updates

lasagne.init


lasagne.nonlinearities

lasagne.objectives

lasagne.regularization

lasagne.random

lasagne.utils

 YOUR AD HERE

Reach over 7 million devs each month when you advertise with Read the Docs.

Sponsored - Ads served ethically

Read the Docs v: latest

Docs » Welcome to Lasagne [Edit on GitHub](#)

Welcome to Lasagne

Lasagne is a lightweight library to build and train neural networks in Theano.

Lasagne is a work in progress, input is welcome. The available documentation is limited for now. The project is on [GitHub](#).

User Guide

The Lasagne user guide explains how to install Lasagne, how to build and train neural networks using Lasagne, and how to contribute to the library as a developer.

- [Installation](#)
 - [Prerequisites](#)
 - [Stable Lasagne release](#)
 - [Bleeding-edge version](#)
 - [Development installation](#)
 - [GPU support](#)
 - [Docker](#)
- [Tutorial](#)
 - [Before we start](#)
 - [Run the MNIST example](#)
 - [Understand the MNIST example](#)
 - [Where to go from here](#)
- [Layers](#)
 - [Creating a layer](#)

Fonte: Lasagne, 2019.

Hyperparameter search

Table 2

Explored hyperparameters and selected values (see Section 4) for SVM, CNN and Data Augmentation.

Method	Hyperparameter	Range	Selected Value	Description
SVM	C	[0.5, 1, 3, 5, 10, 50, 100, 200, 1000]	10	SVM C parameter
	gamma	[50, 20, 14, 10, 5, 2, 1, 0.5, 0.2, 0.1, 0.05]	10	Gaussian kernel gamma
CNN	blockSize	[1-4]	4	Layers per block
	nBlocks	[1-5]	3	Number of blocks
	widthFactor	[8, 16, 32, 48, 64, 96, 128]	16	Multiplicative factor for the number of maps
	log10lr	[-5.0 to -2.0]	-2.43	Learning rate (log10 scale)
	log10wd	[-5.0-0.0]	-1.10	Weight decay (log10 scale)
	batchSize	[16, 32, 48, 64, 96, 128]	128	Samples per minibatch
Data Augm.	zoom	[0.9-1.1]	0.97	Zoom range center
	zoomRange	[0.0-0.25]	0.18	Amplitude of zoom interval
	shear	[0.0-0.35]	0.14	Maximum shear angle (radians)
	wShift	[0.0-10.0]	2.8	Maximum horizontal shift (%)
	hShift	[0.0-10.0]	1.6	Maximum vertical shift (%)
	curves	[0.0-0.75]	0.58	Maximum curve strength
	rot	[0-20]	20	Maximum random rotation (degrees)

Fonte: Uzal *et. al.*, 2018.

Results

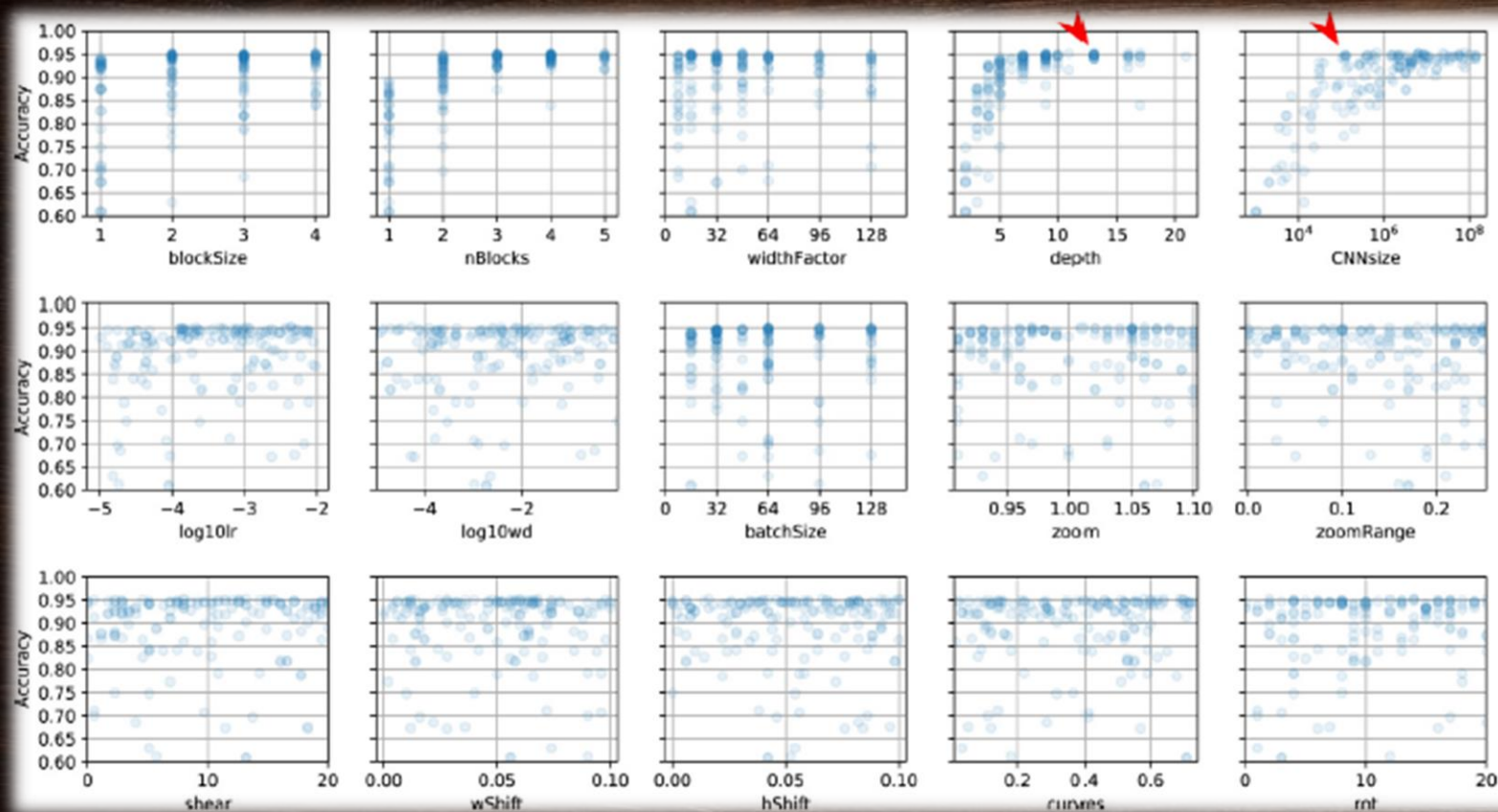
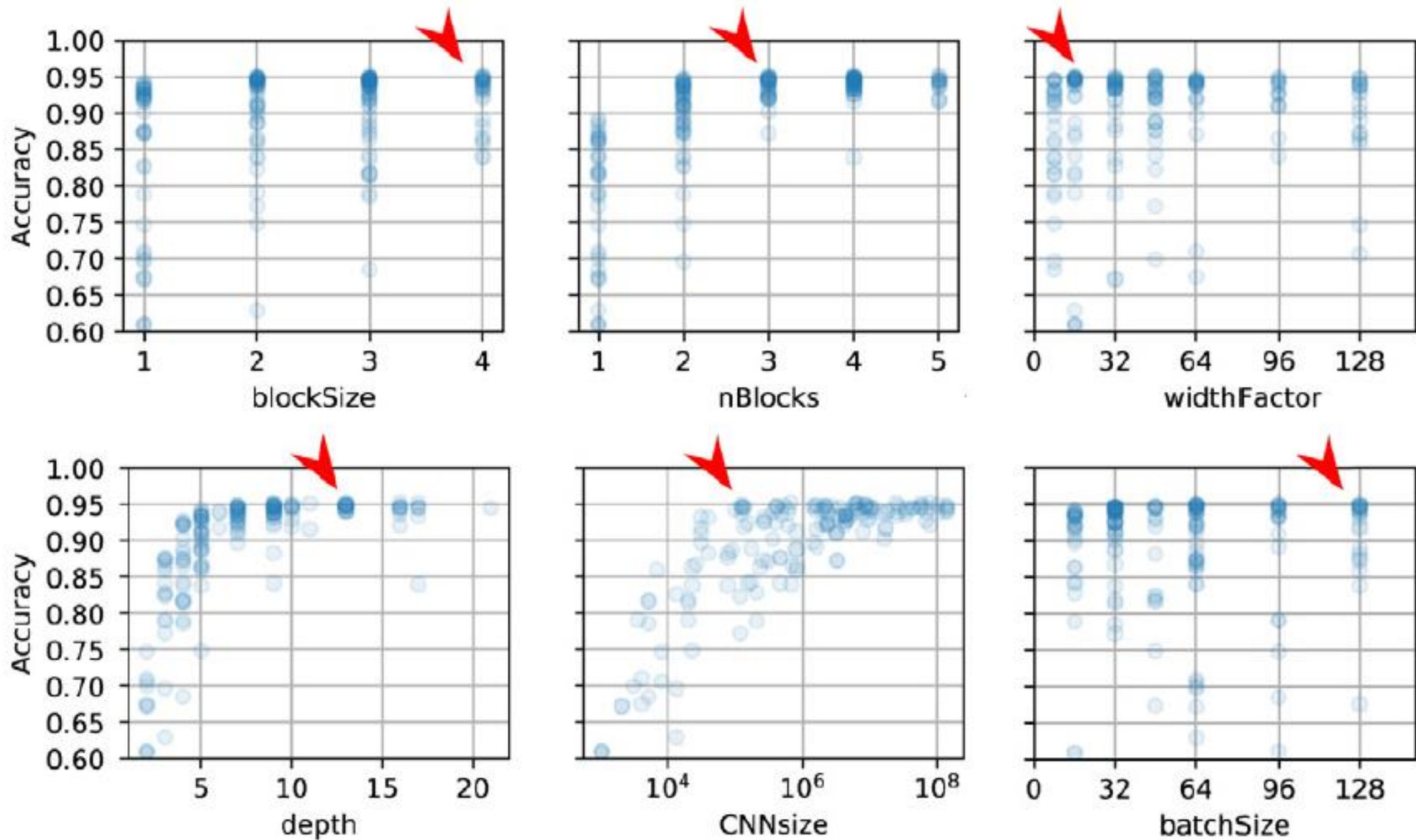


Fig. 5. Random search of hyperparameters for CNN training. The first row corresponds to model hyperparameters (defining network architecture). Variables `depth` and `CNNsize` are significant quantities derived from hyperparameters `blockSize`, `nBlocks`, and `widthFactor`. The rest of the panels corresponds to training algorithm and data augmentation hyperparameters (see Table 2 for details). Validation accuracy is almost insensitive to these training and data augmentation parameters. In order to reach high accuracies (above 90%), the only thing needed is to take a deep enough, high capacity network. Red arrows show the model selected which is a tradeoff between maximizing accuracy and minimizing model size.

Fonte: Uzal *et. al.*, 2018.

Relevant hyperparameters



Results

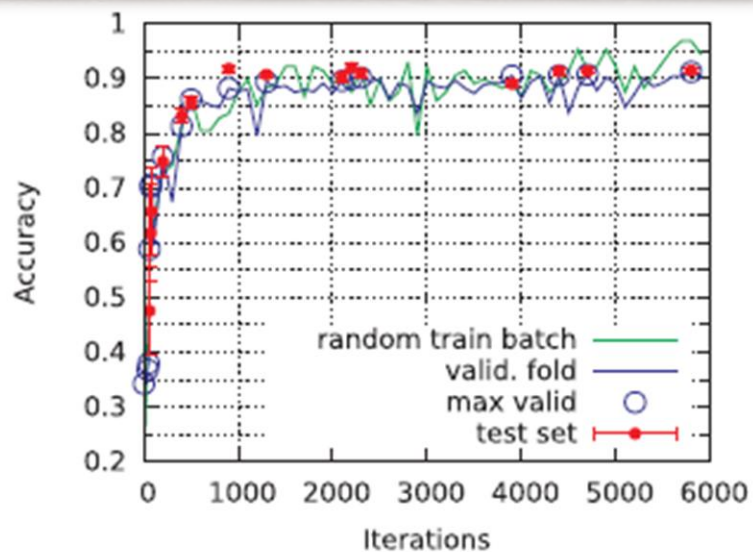
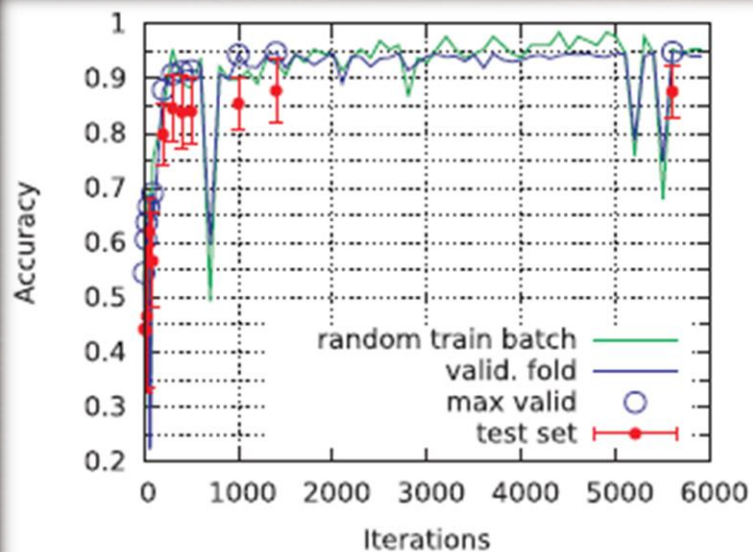
Table 3

Accuracy for different methods trained on Season 1 data and tested on Season 2 data. Mean and deviation for validation accuracies were computed with a group k -fold procedure over training data. Test accuracy mean and standard deviation were computed over test session groups and averaged over k -fold models.

Method	Valid. Accuracy	Test Accuracy
Features + SVM	0.902 ± 0.022	0.504 ± 0.145
CNN without Data Augmentation	0.936 ± 0.009	0.827 ± 0.043
CNN with Data Augmentation	0.951 ± 0.005	0.862 ± 0.052

Fonte: Uzal *et. al.*, 2018.

Results

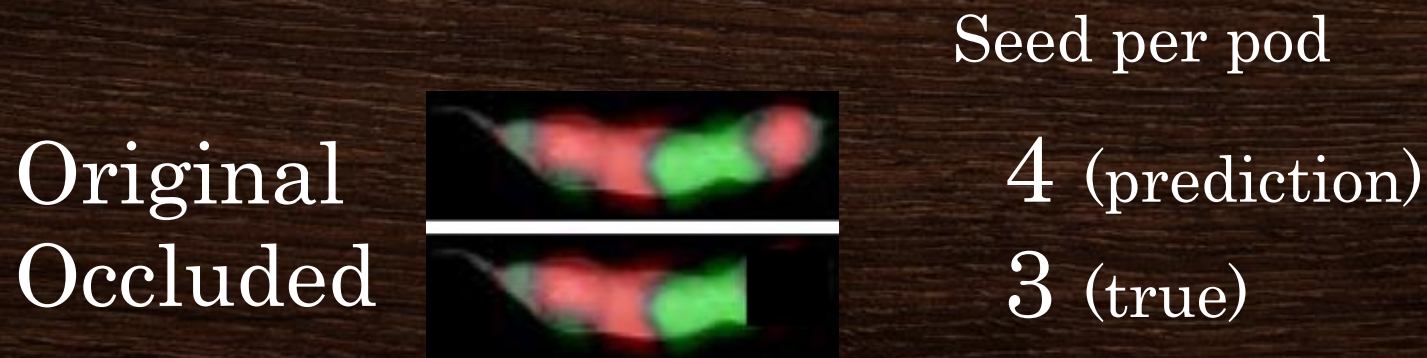
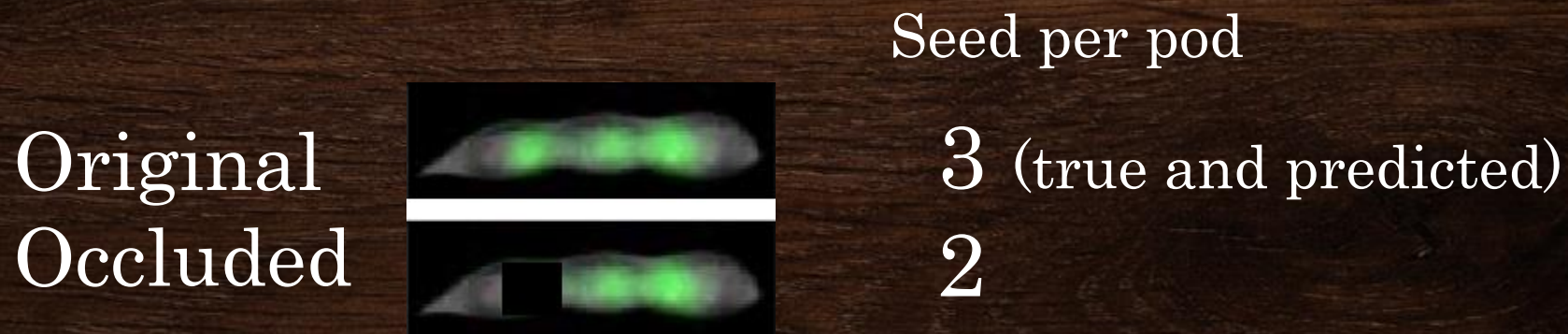


(a) Train set: Season 1. Test set: Season 2 (b) Train set: Season 2. Test set: Season 1

Fig. 6. Training curves for the two different seasons. Green lines: accuracy computed over a random minibatch during training. Blue lines: accuracy computed on the validation fold. Blue empty circles: new maximum in validation accuracy. Red dots: test accuracy. Season 2 dataset appears to contain more diverse and difficult examples which do not exist in the Season 1 version. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fonte: Uzal *et. al.*, 2018.

Detection of important features



Fonte: Uzal *et. al.*, 2018.

Visualizing relevant patterns

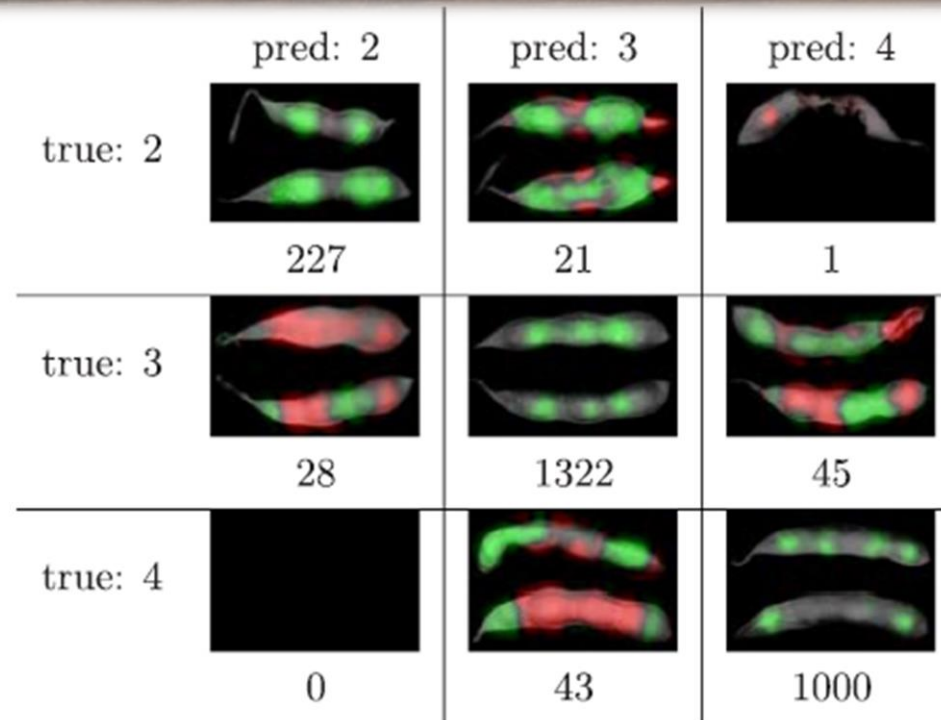


Fig. 7. Confusion matrix with representative samples visualization. Green (red) colored regions indicates regions of positive (negative) correlation with correct class CNN output probability obtained by occlusion experiments. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fonte: Uzal *et. al.*, 2018.

Visualizing relevant patterns

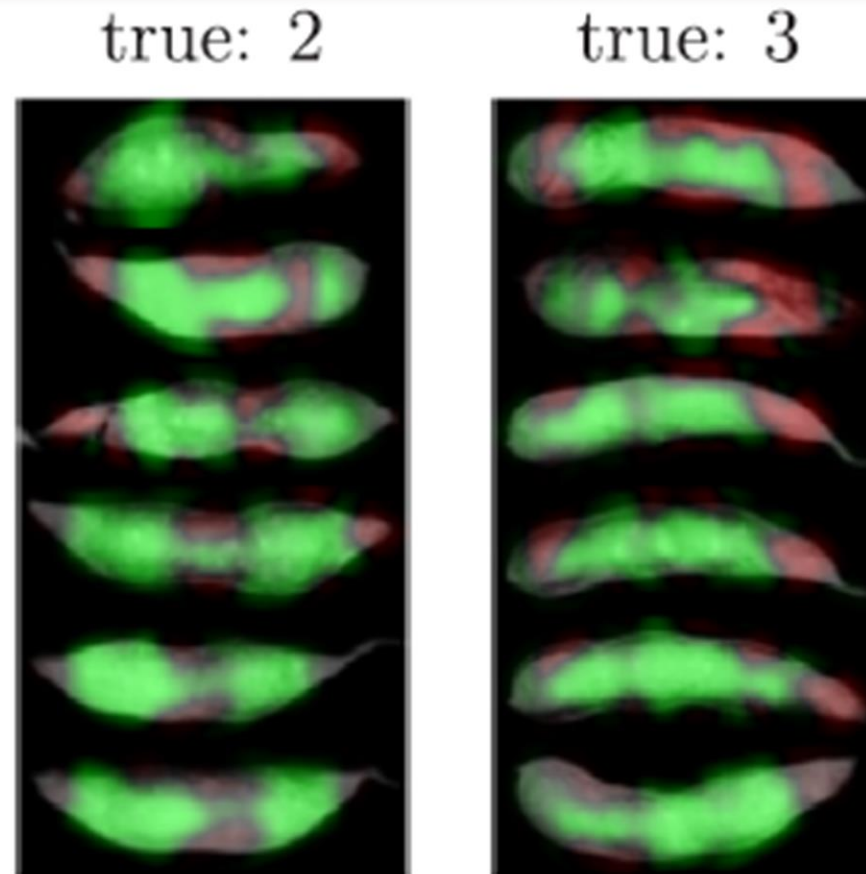


Fig. 8. Visualization of samples where the contour plays an important role. The colors indicate the same as in Fig. 7. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Conclusion and future works

- CNNs have many hyperparameters, but high accuracy can be achieved without precise values
- CNNs outperformed classic approach
- CNNs learnt to detect each seed in the pod
- Detection in field

Referências

- AGRIEXPO. Scanner de fenotipagem de plantas / para estufa /para ambiente interno. Disponível em: <<http://www.agriexpo.online/pt/prod/wps/product-179889-33446.html>>. Acesso em: 20 de Abril de 2019.
- EMPRAPA. Expozebu dinâminca – Embrapa terá maior participação no evento deste ano. Disponível em: <<http://cerradoeditora.com.br/cerrado/expozebu-dinaminca-embrapa-tera-maior-participacao-no-evento-deste-ano/>>. Acesso em: 20 de Abril de 2019.
- GIMP. Gnu image manipulation program. Disponível em: <<https://www.gimp.org/>>. Acesso em: 20 de Abril de 2019.

Referências

- KERAS. Keras: The Python Deep Learning library. Disponível em: < <https://keras.io/>>. Acesso em: 20 de Abril de 2019.
- LASAGNE. Disponível em: <<https://lasagne.readthedocs.io/en/latest/index.html>>. Acesso em: 20 de Abril de 2019.
- MAISSOJA. Qualidade de grãos de soja armazenados em diferentes condições de temperatura e umidade. Disponível em: <<https://maissoja.com.br/qualidade-de-graos-de-soja-armazenados-em-diferentes-condicoes-de-temperatura-e-umidade/>>. Acesso em: 20 de Abril de 2019.

Referências

- OPENCV. The most popular computer vision library just got better. Disponível em: < <https://opencv.org/>>. Acesso em: 20 de Abril de 2019.
- SCIKIT-LEARN. Machine Learning in Python. Disponível em: < <https://scikit-learn.org/stable/>>. Acesso em: 20 de Abril de 2019.
- SISTEMA FAEP. Embrapa explica o abortamento de vagens e enchimento deficiente de grãos em soja. Disponível em: < <https://sistemafaep.org.br/embrapa-explica-o-abortamento-de-vagens-e-enchimento-deficiente-de-graos-em-soja>>. Acesso em: 20 de Abril de 2019.

Referências

- TECHNOLOGYTIMES. Pakistan needs strategy to promote soyabean. Disponível em: <http://www.technologytimes.pk/pakistan-needs-strategy-to-promote-soyabean/>. Acesso em: 20 de Abril de 2019.
- THEANO. Optimizing compiler for evaluating mathematical expressions on CPUs and GPUs. Disponível em: <https://pypi.org/project/Theano/>. Acesso em: 20 de Abril de 2019.
- UZAL, L.C. *et. al.* Seed-per-pod estimation for plant breeding using deep learning. Computers and Electronics in Agriculture, v.150, p. 196-204, 2018.