WADENOW

Release 1 – March 2022

A MATLAB™ free toolbox for early forecasting of the velocity trend of a rainfall-triggered landslide by means of continuous wavelet transform and convolution neural networks applied to rainfall and velocity time series.

Giordano Teza
Università degli Studi di Bologna
Dipartimento di Fisica e Astronomia
Viale Berti Pichat 6/2, 40127 Bologna, Italy
giordano.teza@gmail.com, giordano.teza@unibo.it

Summary

1. Introduction 3
1.1 Aim of the toolbox 3
1.2 Brief description of the method 3
1.2.1 CWT-based scalogram generation 4
1.2.2 CNN-based scalogram classification 4
2. Requirements and installation 7
2.1 Requirements 7
2.2 Conditions - disclaimer 7
2.3 Installation 8
3. Input data preparation and preliminary computations 9
3.1 Rainfall data 9
3.2 Velocity data 10
3.2.1 Position data processing for training 10
3.2.2 Position data processing for CNN operation 11
3.2.3 Velocity computation 12
4. Computations 13
4.1 Overview and general setting 13
4.2 General script 17
4.3 Rainfall and velocity time series inspection 18
4.4 Scalogram computation and image generation 20
4.5 Velocity trend classification 22
4.6 Data homogenization and augmentation 23
4.7 Partition into training, validation and test datasets 26
4.8 Transfer learning or training resume 26
4.9 Operational use of the trained CNN 29
5. List of developed MATLAB functions 34
5.1 Main functions and scripts 34
5.2 Ancillary functions 35
5.3 External MATLAB functions 35
5.4 Sample data for tutorial purposes 35
6. References 36

1. Introduction

1.1 Aim of the toolbox

WADENOW (WAvelet- and Deep learning-based NOWcasting of landslide kinematics) is a collection of MATLAB scripts. It is the implementation of a procedure aimed at short term (from some hours to a few days) forecasting the behavior of an unstable slope, based on Continuous Wavelet Transform (CWT) and a Convolutional Neural Networks (CNN). Sufficiently long (some years) time series of rainfall, provided by a pluviometer, and velocity, provided by a real-time or at least near-real-time monitoring system, are required. The procedure is fully described in Teza et al. (2022) and no more than the main concepts are shown here.

For each evaluation time, the rainfall and velocity time series related to several previous days (e.g. 15 d) are extracted and the corresponding scalograms are computed by means of CWT. A CNN classifies the scalograms taking into account seven kinematical levels:

- L_1: low velocity;
- L_2: transition from low to mid velocity. If the CNN is coupled to an early warning system, this condition can lead to the automatic emission of a pre-alarm signal (the terms "pre-alarm" and "alarm" are not unanimously used, see e.g. Guzzetti et al., 2020 for possible alternative terms);
- L_3: mid velocity;
- L_4: transition from mid to high or also extreme velocities. In some cases, a transition to high/extreme velocity could also directly starts from a low level. Regardless to the start level, such an output could lead to the automatic emission of an alarm signal;
- L_5: high velocity;
- L_6: transition from high/extreme to mid velocity, with the possible non-automatic emission of an alarm reset signal;
- L_7: transition from mid to low velocity, with the possible non-automatic emission of a pre-alarm reset signal.

If the available time series are not enough large to allow the scalogram classification with seven possible outputs, a classifier with four possible outputs could be used:

- L'_1: low velocity;
- L'_2: transition from low to high or also extreme velocity;
- L'_3: high velocity;
- L'_4: transition from high/extreme to low velocity.

In this way, useful data about the start, continuation and end of a critical period can be provided to the decision makers, fact which is very important if the unstable slope directly or indirectly threatens residential areas or infrastructures.

1.2 Brief description of the method

The proposed procedure in based on:

(1) CWT-based scalogram generation;
(2) CNN-based scalogram classification.

1.2.1 CWT-based scalogram generation

Let $x = x(t)$ be a 1-D time series. Its CWT with a mother wavelet ψ is

$$CWT(a,b,x,\psi) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} x(t) \bar{\psi}\left(\frac{t-b}{a}\right) dt \,, \tag{1.1}$$

where $a,b \in \mathbb{R}, a \neq 0$ and $\bar{z}$ is the complex conjugate of a $z \in \mathbb{C}$. The mother wavelet ψ must be a zero average function which oscillates and decays, well localized both in time and frequency. In this way, $CWT(a,b,x,\psi)$ measures the variation of $x(t)$ in a neighborhood of *b*, called shift factor, whose size is proportional to *a*, the scale factor. Unlike Fourier transform, CWT can represent non-stationary signals.

The Morlet wavelet (Goupilaud et al., 1984) is particularly suitable for detection and analysis of transient signals and, therefore, is largely used as mother wavelet for CWT of geophysical time series, including climatological and meteorological data (Hochman et al., 2019). The complex Morlet wavelet with center frequency f_0 and time decay parameter σ_t^2 is

$$\psi(t) = \frac{1}{\sqrt{2\pi}\,\sigma_t} \exp(i2\pi f_0 t) \exp\left(-\frac{t^2}{2\sigma_t^2}\right), \tag{1.2}$$

where f_0 is the global maximum of the Fourier transform of $\psi(t)$ and σ_t^2 is such that (Cohen, 2019)

$$\sigma_t = \frac{n}{2\pi f_0} \tag{1.3}$$

and *n* is the number of cycles. This latter parameter controls the decay in the time domain and the corresponding energy spread in the frequency domain. In particular, if *n* increases, the CWT energy is more concentrated around f_0, and the wavelet decay in the time domain becomes slower, with more oscillations of the sinusoid. The choice of *n*, and therefore of σ_t^2, depends on specific application.

The scalogram is a representation of the signal energy computed using the modulus of the CWT. It is the equivalent for the CWT of what the spectrogram is for the Fourier transform. The shift factor *b* directly comes from the length of the time series. Let *t** be a generic evaluation time, i.e. a time for which the data are to be analyzed and a forecast should be provided, expressed in days. The time span from t^*-N_A to t^*+N_B is considered, where N_A can be, e.g., 15 d. As for N_B, in the case of a rainfall time series a value such that $0 < N_B \leq 1-2$ d has two justifications:

(i) it contributes at a reduction of the problems connected to edge-effects (Lilly, 2017; Addison, 2020), which are particularly significant if sudden heavy rainfalls, which could affect the landslide kinematics, occur shortly before *t**;

(ii) reliable weather forecasts are normally available on a 1-2 d scale and can be integrated into the forecasting system.

A similar justification cannot be proposed in the case of a velocity time series because it makes no sense in the normal operation of the forecasting system.

1.2.2 CNN-based scalogram classification

The scalogram classification is carried out by using a pre-trained CNN, e.g. a VGG19 model (Simonyan and Zisserman, 2015), repurposed by means of transfer learning. A CNN consists of two parts:

(1) convolutional base, aimed at generating some features from the input image and composed by a stack of convolutional and pooling layers;
(2) classifier, aimed at classifying the image on the basis of the features detected by the convolutional base and usually composed by fully connected layers.

The transfer learning is carried out by replacing the original classifier with a new classifier that fits the new classification purposes, and fine-tuning of the CNN by means of a training with suitable input data. In this way, the pre-trained model is repurposed in accordance with the new requirements (Weiss and Khoshgoftaar, 2016).

The input images for the CNN training and operation represent scalograms and the outputs describe the velocity trends. Two configurations are proposed:

(i) the input images are generated by using both rainfall and velocity data. In this case, the velocity data are used for both scalogram computation and trend evaluation;
(ii) the input images are generated by using rainfall data only. In this case, the velocity data are used for trend evaluation only (Figs. 1.1.c and 1.1.d). If the topographic monitoring system is temporarily out of order or off-line, this fact does not affect the forecasting system. However, besides this advantage, the configuration (ii) has a significant disadvantage because only information on the phenomenon that triggers the motion of the landslide, which is an indirect indicator of instability, is used to provide forecasts.

The configuration (i) and (ii) are not mutually exclusive, but could be advantageously used together. The CNN trained according to the approach (i) is used to provide forecasts in standard conditions, where the second CNN is used to provide forecasts in those cases where the monitoring system is temporarily unavailable.

The input image size must be compatible with the used CNN model, 227×227×3 in the case of VGG19. For the configuration (i) each image shows two scalograms. If $N_B > 0$, the alignment of the scalograms is kept by means of zero-padding of the velocity one. Examples of input images for this configuration and each level are shown in Fig. 1.1.

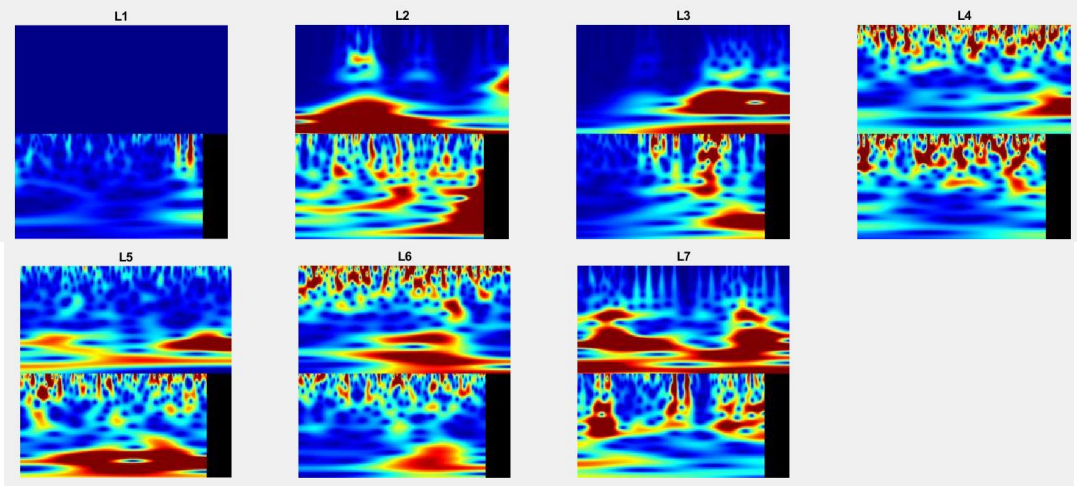

Figure 1.1 Examples of images with rainfall (upper sector of each image) and velocity scalograms for the seven levels L_1-L_7.

For the configuration (ii), each image shows a rainfall scalogram. Examples of input images for this configuration and each level are shown in Fig. 1.2.

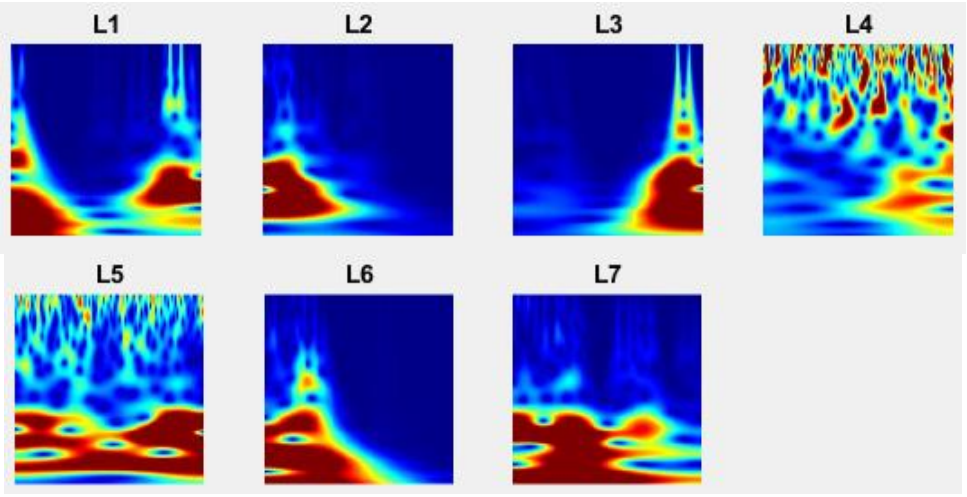

Figure 1.2 Examples of images with rainfall scalograms for the seven levels L_1-L_7.

2. Requirements and installation

2.1 Requirements

WADENOW consists of several MATLAB functions and scripts whose complete functionality requires:

- MATLAB 2018a, or later releases;
- MATLAB Wavelet Toolbox™;
- MATLAB Deep Learning Toolbox™ (DLT);
- One or more DLT models of pretrained CNNs. The model of a CNN support package can be downloaded and installed by typing its name on the MATLAB command window (MCW), i.e.

```
alexnet
vgg16
vgg19
googlenet
resnet18
```

2.2 Conditions - disclaimer

The package is free and can be used if the following legal requirements are satisfied:

a) **any author's responsibility for the use of WADENOW is excluded, according to the disclaimer (see below)**;
b) WADENOW is used for scientific purposes only;
c) if one or more articles are published using data processed by means of WADENOW, the author of this package is cited and the paper where the method is shown is cited (please see References, Teza et al., 2020);
d) any commercial use of WADENOW is excluded.

DISCLAIMER: The author of WADENOW toolbox accepts no responsibility for damages resulting from the use of this product and makes no warranty or representation, either express or implied, including but not limited to, any implied warranty of merchantability or fitness for a particular purpose. This software is provided "AS IS", and the user assumes all risks when using it. Please also note that the operation function (see Subsection 4.9) must be customized by the user. The responsibility on each conclusion based on data processed by means of WADENOW exclusively relies on the user.

Please communicate any problem encountered in use of WADENOW toolbox and any suggestion aimed to improvement of the software performance and portability (please send an e-mail to giordano.teza@gmail.com).

Collaboration aimed at improving the toolbox is warmly encouraged.

2.3 Installation

The WADENOW.zip file contains all the files of the package. Some sample files are also included for tutorial purposes. If the package is used as an autonomous toolbox for image preprocessing, registration an mosaicking, the files should be extracted in a directory (named e.g. wadenow) subdirectory of the MATLAB work directory. All the scripts are m-files, and their codes and helps therefore are completely accessible to the user and could be used to translate them into Python. All the scripts are platform independent, i.e. can run under Windows, Unix and MacOS.

If Windows is the operating system (OS) and C:\Users\HJ\Documents\MATLAB is the work directory, the files can be extracted in C:\Users\HJ\Documents\MATLAB\WADENOW.

In order to call the package whichever is the actual MCW folder, two options are possible: startup file and set path. A startup file can be written by the user. If Windows is the OS and WADENOW is placed in the folder C:\Users\HJ\Documents\MATLAB\WADENOW, the row

```
addpath C:\Users\HJ\Documents\MATLAB\WADENOW; % addpath row
```

should be added to startup.m script. The functional form of this command line, i.e.

```
addpath('C:\Users\HJ\Documents\MATLAB\WADENOW'); % addpath row
```

is also allowed.

Please note that Unix/Linux/MacOS require '/' instead of '\'.

If a file named startup.m is placed in the MATLAB directory (for example, C:\Program Files\MATLAB\R2018a\bin in the case of R2018 in Windows), it is automatically executed at each MATLAB start. In this way, all the defined search paths are automatically added at each MATLAB start and the folder change to use WADENOW is unnecessary. The WADENOW functionalities do not depend on user's choice about the startup. More information about addpath function in a startup file can be found in http://www.mathworks.it/it/help/matlab/ref/ addpath.html.

Another option is the use of the Set Path dialog box, which appears by typing

```
pathtool
```

on the MCW or by selecting Set Path in Home menu of MATLAB desktop. The button Add Folders allows the choice of the folder and other buttons allow the choice of the folder order for the search of the files. Please see http://www.mathworks.it/it/help/matlab/matlab_env/using-the-matlab-search-path.html to have more information about the Set Path dialog box.

See also Subsection 4.9 about the startup file, in particular for the automatic call of the script that performs the CNN operation aimed at providing forecasts.

3. Input data preparation and preliminary computations

The input data for both the training stage and the operation stage are rainfall and velocity time series (if scalograms are computed with rainfall data only, rainfall data only are used in the operation stage). Therefore, time series suitable for these computations are necessary.

The solution to the problem of extracting data and transforming it into the form necessary for calculations with `WADENOW` depends on the characteristics of the specific monitoring system used. Therefore, the user may need to write specific MATLAB functions. However, this chapter shows an example for both rainfall and kinematic data.

3.1 Rainfall data

As described in Section 4.2, the function `PluVelInspect` requires, among the input data, the matrices `MR1=[tR R1]` and `MR7=[tR R7]` of 1 d and 7 d cumulative rainfall time series with a sampling rate of 24 cpd (cycles per day), where the time column vector `tR` is expressed in form of MATLAB date serial number. Nevertheless, a pluviometer generally provides an ASCII `.txt` file which can easily be changed to have an Excel `.xls/.xlsx` file. A section of a typical file provided by a pluviometer which operates with a sampling rate of 96 cpd (four data per hour) is like the one in the left column, whereas the corresponding file translated to `.xlsx` is like the one in the right column:

```
2016-06-24 17:00:00,0
2016-06-24 17:15:00,0
2016-06-24 17:30:00,0
2016-06-24 17:45:00,0
2016-06-24 18:00:00,0
2016-06-24 18:15:00,0
2016-06-24 18:30:00,0
2016-06-24 18:45:00,0
2016-06-24 19:00:00,2.2
2016-06-24 19:15:00,0.4
2016-06-24 19:30:00,0
2016-06-24 19:45:00,0
2016-06-24 20:00:00,0
```

7	24/06/2016 17:00	0		
8	24/06/2016 17:15	0		
9	24/06/2016 17:30	0		
10	24/06/2016 17:45	0		
11	24/06/2016 18:00	0		
12	24/06/2016 18:15	0		
13	24/06/2016 18:30	0		
14	24/06/2016 18:45	0		
15	24/06/2016 19:00	2.2		
16	24/06/2016 19:15	0.4		
17	24/06/2016 19:30	0		
18	24/06/2016 19:45	0		
19	24/06/2016 20:00	0		

Section of a typical file provided by a system including a digital pluviometer in ASCII form (left) and in Excel form (right)

A file in Excel form is often used to prepare the data before the training. The data processing carried out in the CNN operation must be carried out in an entirely automatized way and, therefore, an ASCII file must be directly used.

In order to obtain a MATLAB matrix from such an Excel file or an ASCII file, the function **`extraPluvio`** can be used:

`M=`**`extraPluvio`**`(FILENA,Delimiter,date1,date2)`

Input data:

`FILENA`: name of the input ASII or `xls/xlsx` file. If `FILENA` is undefined or empty, the filename is interactively managed. The columns of the input data are

```
[date value]
```

where `date` is in one of the following forms:

```
yyyy-mm-dd HH:MM:SS
yyyy-mm-dd HH:MM
yyyy-mm-dd
dd/mm/yyyy HH:MM
dd/mm/yyyy
```

and `value` are numerical data.

`Delimiter`: delimiter of the input ASCII file (of course, such a variable is active only if the input file is ASCII). If undefined or empty, `Delimiter=','` is used.

`date1, date2`: lower and higher date limits. If date1 (date2) is undefined or empt are the ones that allow the generation of the required matrix starting from data provided by a robotic total station which acquire a series of artificial targets (corner cubes) placed on the unstable zone with sampling rate of 4-24 cpd (the reference targets and the targets characterized by higher velocity are acquired with 24 cpd sampling rate. The other targets are acquired four times per day). These functions, which are described in this section, can be adapted by the user to its specific case.

3.2 Velocity data

3.2.1 Position data processing for training

In case of data processing for training, Excel input data are generally used.

The function **`extraTarget2cell`** allows the generation of a cell variable from total station data

`C=`**`extraTarget2cell`**`(strCell,CommonPart,ext)`

This function takes the position data related to some targets of a total station, where `strCell` is a cell variable of strings with the target names, and `CommonPart` is a string of the common part of the input filenames. The data related to the k-th target are taken from the file whose name is `[CommonPart strCell(k) ext]`.

The output cell `C` is such that `C{k,1}` is `strCell(k)`, i.e. the name of the k-th target, and `C{k,2}` is the data matrix, according to the input file.

If `strCell` is undefined or empty, this cell is used:

```
strCell ={'P1','P2','P3','P4','P5','P6','P7','P8','P9','P10',…
   'P11','P12','P13','P14','P15','P16','P17','P18','P19',…
   'P20','P21','P22','P23','P24','P25','P26','P27','P28','P29',
   'P30','P31',...
   'N1','N2','N3','N4','N5','N6','N7','N8','N9','N10',...
   'GPS_ROV 1','GPS_ROV 2',...
   'R1','R1_BIS','R2','R2BIS','R3','R4','R5'}.
```

If `ext` is undefined or empty, `ext=.xlsx` is used.

The default option for `strCell` is related to the specific case of Perarolo di Cadore landslide, described in Teza et al. (2020).

The next step is the generation of a position array suitable for computations. This is carried out by means of **`cell2posArray`** function:

`M=`**`cell2posArray`**`(C,date1,date2)`

This function extracts the topographical data from the `N-by-2` cell variable `C`, where `N` is the number of targets, `C{k,1}` is the position matrix of the k-th target and `C{k,2}` is the corresponding name.
The input variables date1 and date2 are the initial and final dates of the output array, whose data are hourly, i.e. a 24 cpd (cycles per day) sampling rate is considered for each target, regardless to the real sampling rate. The dates date1 and date2 are intended in serial date number form.
The data related to the k-th target are placed in the matrix `M(:,:,k)`.
For the targets characterized by sampling rate lower than 24 cpd, the positions for which no data are provided are `NaN`.

3.2.2 Position data processing for CNN operation

In the case of position data processing for CNN operation, the file must be managed in an entirely automatic way. For this reason, the ASCII file provided by the surveying system must be directly managed. The function that performs this is for the extraction of topographical data provided by a total station is **`extraRTS2array`**:

`M=`**`extraRTS2array`**`(FILENIN,Delimiter,strCell,N)`

This function extracts the topographical data from the ASCII file `FILENIN`, assuming that they are in the form

```
[Date Reflector Angle1 Angle2 Distance x y z],
```

where `Date` is the acquisition date in the form `dd/mm/yy HH:MM`, `Reflector` is a string consistent with the input variable `strCell`, `Angle1`, `Angle2` and `Distance` are the polar coordinates and `x`, `y`, `z` are the Cartesian coordinates. If `FILENIN` is undefined or empty, the filename can be interactively managed.

`Delimiter` is the delimiter of the ASCII file. If it is undefined or empty, `Delimiter='\t'` (tabulation) is used.

The input `strCell` is a cell variable of strings with the target names. If `strCell` is undefined or empty, this cell is used:

```
strCell ={'P1','P2','P3','P4','P5','P6','P7','P8','P9','P10',…
   'P11','P12','P13','P14','P15','P16','P17','P18','P19',…
   'P20','P21','P22','P23','P24','P25','P26','P27','P28','P29',
   'P30','P31',...
   'N1','N2','N3','N4','N5','N6','N7','N8','N9','N10',...
   'GPS_ROV 1','GPS_ROV 2',...
   'R1','R1_BIS','R2','R2BIS','R3','R4','R5'}.
```

The number N is the number of last considered rows. Is undefined or empty, it is N=36000.

The data related to the k-th target are placed in the matrix M(:,:,k) of the output 3D array M. The sampling rate is assumed to be 24 cpd. For the targets characterized by sampling rate lower than 24 cpd, the positions for which no data are provided are NaN.

3.2.3 Velocity computation

The subsequent step is the generation of a velocity array. The corresponding function is **velArray**:

MV=**velArray**(M,NHSTEP,NHM,VCOL)

Computes the velocity array MV from the position array M, with a time step of NHSTEP hours and mean on NHM steps by computing the least square linear fit in the time span [tc-NHSTEP*NHM, tc] for each computation time tc, where

MV(:,:,k)=[tc x y z vx vy vz].

The 3-components vector VCOL indicates the columns of the array M to be used to compute the velocity. For the k-th target, the velocity is therefore computed on the basis of M(:,VCOL,k) data. If VCOL is undefined or empty, the last three columns of each matrix M(:,:,k) are considered.

The last step is the computation of the time series of mean velocity which represent the significant landslide kinematics. The function **selectMeanVelocity** is used to perform this. Syntax:

MV=**selectMeanVelocity**(MVIn,IDV,d1,d2)

A velocity time series MV, in terms of absolute velocity, is computed for each time in MVIn(:,1,1) in the range from d1 to d2 and taking the mean from the targets whose indices are in the index set IDV.
If d1 (d2) is undefined or empty, d1=MV(1,1,1) (d2=MV(end,1,1)) is used.

The matrix MV provided by selectMeanVelocity is the variable to be used for CNN training, as shown in Chapter 4.

Some real data, provided by a pluviometer and a robotic total station, are added to the WADENOW toolbox for tutorial purposes. The script **PluVelExample** allows their management, leading to the time series MR1, MR7 and MV. In order to run this script, the user should type

PluVelExample

on the MCV. The user can change the parameters by direct acting on this script.

4. Computations

4.1 Overview and general setting

The **WADENOW** functions and scripts are developed in order to allow these steps:

1) Rainfall and velocity time series inspection and possible cleaning of false velocity peaks, i.e. peaks not due to rainfall. This initial step is not compulsory and could be carried out before the execution of `WADENOWGeneral`. The involved function is `PluVelInspect`;
2) Scalogram computation for rainfall and velocity time series or rainfall only time series. A menu box allows the choice between rainfall and velocity scalograms and rainfall only scalograms. The function is `PluVelScalogram`;
3) Trend classification of the basis of user-defined thresholds. The function is `trendClass`;
4) Data homogeneization and augmentation is order to allow training with balanced datasets. The function is `dataHomAug`;
5) Scalogram images partition into training, validation ad test Datasets. The function is `dataTVT`;
6) Transfer learning of an available trained CNN model, chosen among alexnet, VGG16, VGG19, googlenet, resnet18, or training resume, with the scalogram images. The function is `TRLearnPluVel`. Please note that:
   (a) the transfer learning of a CNN model requires the installation of the corresponding support package (a warning message is shown if an unavailable model is called);
   (b) in case of training resume, the function `TRLearnPluVel` should be directly used (see below).
7) Operational use of the CNN to provide forecasts. Such an use requires different approaches depending on the monitoring system. Anyway, the function `PluVelOpera` is provided as an example.

As depicted in Chapter 3, time series of rainfall and velocity are the input data. In the case of rainfall, the data are provided by a rain gauge and a single time series is obtained for each landslide. In the case of velocities, the studied unstable slope could be subdivided into several kinematic areas that should be monitored. In this case, more than one velocity time series are obtained and the procedure explained below must be applied to each velocity time series (or, better, each couple of rainfall and velocity time series).

A structure array defined at the beginning of the calculation process allows the definition of all the main options (type of wavelet and parameters necessary for the generation and classification of the scalograms and for the training of the neural network). It is called, like an object, by all the `WADENOW` main functions. The user can choose between the default analytic Morlet wavelet, characterized by $\omega_0 = 6$ d^{-1} and $\sigma_t = 1$ d, and a generic analytic Morlet wavelet. In order to allow this choice, the functions provided by Erickson (2020), which are modified versions of the ones provided by Torrance and Compo (1998), are added to `WADENOW` after further modifications, carried out in accordance with Mallat (2009), to allow the choice of σ_t (Erickson, 2020 allows the choice of ω_0 only).

The structure array is `ParamWN` and can be managed in two ways:

(i) by acting on the **`DefParam.m`** file. In this case, the user directly changes the input parameters on the lines of such a function;

(ii) by using the **`DefParamInteractive`** function, with the input dialog box shown in Fig. 4.1.

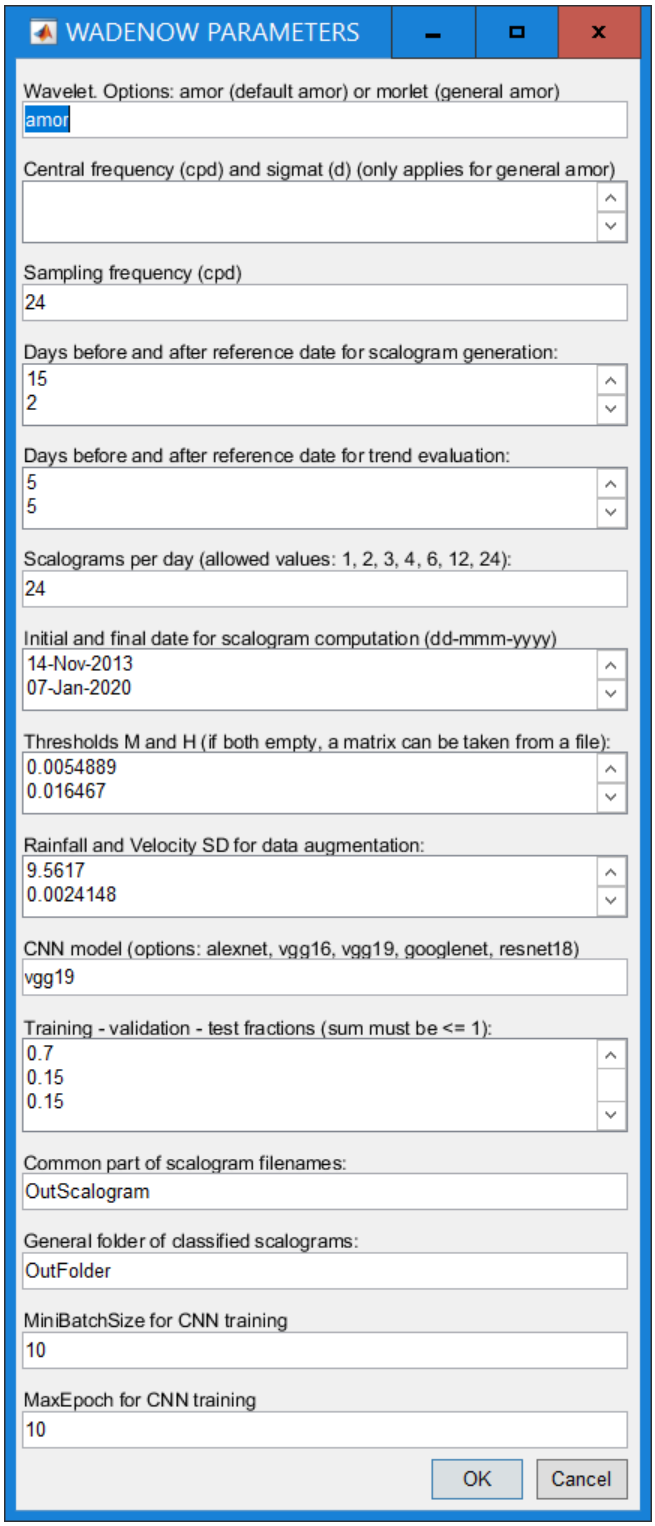

Figure 4.1 Parameters of `ParamWN` managed by means of an input dialog box which appears is `DefParamInteractive` is used. Note that the thresholds for the trend classification can be managed by means of a MATLAB file.

The generated structure variable can be saved, with its name, in a MATLAB `.mat` file.

Regardless to the used approach for the parameter management, `ParamWN` is a struct variable whose fields are:

`Wavelet`: wavelet. Possible values: `amor` (MATLAB default) and `morlet` (analytic morlet whose Parameters can be user defined). If a string different from `amor` and `morlet` is chosen, `amor` is used.

`F0`: central frequency (expressed in cpd) of the complex oscillation (default: `6/(2*pi)` cpd, to have `omega0=6` d^{-1}).

`Sigmat`: Parameter of the Gussian bell with tapers the complex oscillation (default: 1 d).

Note: the Parameters `F0` and `Sigmat` are active only if the option `morlet` is chosen.

`Fs`: sampling frequency (24 cpd, i.e. 1h sampling time)

`VoicesPerOctave`: voices per octave, i.e. number of intermediate scales in an octave, i.e. a frequency doubling (note that this field is not shown in case of interactive choice).

`Na`: the starting date of the scalograms related to `t(k)` is `t(k)-Na`, where `Na` is expressed in days, under the condition `t(k)-Na >= t(1)` (if this condition is not fulfilled, the corresponding scalogram is not computed).

`Nb`: the ending date of the pluviometric scalogram related to `t(k) is t(k)+Nb`, where `Nb` is expressed in days, under the condition that `t(k)+Nb <= t(end)` (if such a condition is not fulfilled, the corresponding scalogram is not computed). The ending date of the velocity scalogram always is `t(k)`.

`Nc`: number of days to be considered before `tRef`. If it is empty, it is `Nc=5`;

`Nd`: number of days to be considered after `tRef`; If it is empty, it is `Nd=5`;

`nPerDay`: number of scalograms per day. The allowed values are 1 (one scalogram per day at 0h), 2 (two scalograms at 0h and 12 h), 3 (0h, 8h, 16h), 4 (0h, 6h, 12h, 18h), 6 (0h, 4h, 8h, 12h, 16h, 18h, 20h) 12 (even hours), 24. If `nPerDay` is undefined, empty or not an allowed value, `nPerDay = 1` is used.

`DateIn`: initial date for which the scalograms are computed (this date must be expressed as standard MATLAB serial number). The start time of the first computed scalogram is `max(d1-Na,t(1))`. If `DateIn` is undefined of empty, it is the initial date of the time series.

`DateFin`: final date for which the scalograms are computed (this date must be expressed as serial number). The end time of the last computed scalogram is `min(d2+Nb,t(end))`. If `DateFin` is undefined of empty, the final date is the last date of the time series.

`Vmh`: scalar/vector/matrix for the management of thresholds for the segments classification. Options about such an argument:

- `Vmh` is a scalar. in this case, a 4-levels classification (including the transitions) is carried out, according to:

  `1: L, 2: L->H, 3: H, 4: H->L,`

with the threshold `VH=Vmh`.

- `Vmh` is a 2-elements vector. In this case, a 7-levels classification is carried out with the thresholds `VM=Vmh(1)` and `VH=Vmh(2)` (the vector is sorted in ascending order). The levels `L`, `M` and `H` are characterized by the conditions `V<=M`, `M<V<=H` and `V>H` respectively. The outputs are:

  `1: L, 2: L->M, 3: M, 4: M->H (or also L->H),`
  `5: H, 6: H->M, 7: M->L (or also H->L).`
- `Vmh` is a 2-column matrix `Vmh=[t VH]`, where `VH(h)` is the time-dependent threshold at `t(h)`, as in the first case (four level output).
- `Vmh` is a 3-column matrix `Vmh=[t VM VH]`, where `VM(h)` and `VH(h)` are the time-dependent thresholds at `t(h)`, as in the second case.

If `ParamWN` is interactively managed by means of an input dialog box and the thresholds must be defined by means of a matrix, this matrix is taken from a file which must have the field `Vmh`.

`sigmaDA`: standard deviation of rainfall and velocity to be used in data augmentation. Options about `sigmaDA`:

- vector having two elements. In this case, it is `sigmaR=sigmaDa(1)` and `sigmaV=sigmaDA(2)`.
- Scalar. In this case, it is `sigmaR=sigmaDA`. Please note that `sigmaDA` must be coherent with the `sigIn`.
- Undefined or empty. In this case, `sigmaDA`, can be managed in an interactive way.

`CNNmodel`: allowed strings: `alexnet`, `vgg16`, `vgg19`, `googlenet`, `resnet18`. If no a correct string is chosen, the default choice `vgg19` is used.

`SizeIm`: image size. Such a field is automatically defined on the basis of the chosen `CNNmodel`.

`VectPart`: 3-elements vector such that `VectPart(1)` is the fraction of images for training, `VectPart(2)` is the fraction for validation and `VectPart(3)` is the fraction for test. The condition `sum(VectPart)<=1` must be satisfied. If `VectPart` is undefined or empty, or the condition about `sum(VectPart)` is not satisfied, the default input value `VectPart=[0.7 0.15 0.15]` is used;

`ComPart`: common part of the output filename (folder name and constant part of the filename). Each filename is completed with the date string. For example, if `ComPart` is the string `pluvioData\LongBeach15days`, the filename of the scalogram of data related to the 15 d until the date 15 march 2020, 12:00:00 is the string `pluvioData\LongBeach15days_15-Mar-2020-120000.jpg`. If `ComPart` is undefined or empty, `ComPart=''` (empty string) is used.

`FoldOut`: general folder of output data. For each level `Lk` (`k=1:4` or `k=1:7`, depending on the number of classes represented in `ClassTS`, last column vector of the variable `L` provided by `trendClass` function, a nested folder whose name is `Lk` is

generated and the corresponding output scalogram images are placed here. Clearly, possible undefined time series segments do not lead to images of the output datastore.

`MiniBatchSize`: Parameter for CNN training (default: 10).

`MaxEpoch`: Parameter for CNN training

Among the parameters to be chosen there are the two velocity thresholds (a threshold only in the four outputs case). At the time *t*, let $v_S(t)$ be the velocity provided by the monitoring system. The landslide velocity is assumed to be low, mid or high if $v_S(t) \le V_M$, $V_M < v_S(t) \le V_H$ or $v_S(t) > V_H$ respectively, where the thresholds V_M and V_H are chosen according to the general behavior of the unstable slope and, if applicable, the results of numerical modeling. These thresholds can be constant or time-dependent values. For example, it could be $V_M(t) = F_M v_{S6m}(t)$ and $V_H(t) = F_H v_{S6m}(t)$, where $v_{S6m}(t)$ is the mean velocity in this area in the last 6 months, excluding the periods of heavy rainfall, and F_M, F_H are chosen multiplicative factor. Other choices are possible. In the case of interactive choice and thresholds variable with the time, these thresholds are managed by means of an external file (see above about `Vmh`).

The parameter structure variable `ParamWN` can be placed on the MCW by typing

```
ParamWN=DefParam
```

(in this case, the data related to the current `DefParam` setting are used to generate `ParamWN`)

or, in case of interactive choice,

```
ParamWN=DefParamInteractive
```

4.2 General script

`WADENOWGeneral` is the general script for the Continuous Wavelet and Deep Learning based rainfall/velocity time series classification for nowcasting purposes. This scripts allows the guided execution of the steps 1)-6) described in Section 4.1:

The user can active the guided approach to scalogram generation and classification by typing

```
WADENOWGeneral
```

on the MCW.

A menu box allow the choice of the inclusion of the procedure of time series inspection/cleaning (Fig. 4.1.a). If such a procedure is included, two files must be managed:

- a file with the cumulative rainfall data at 1 d and 7 d. **Such a file must contain at least two variables whose fields must necessarily be `MR1` and `MR7`**;
- a file with the velocity data, which must contain at least a variable whose field is `MV`.

These variables must necessarily be coherent with the required arguments of `PluVelInspect` function (see Section 4.3). If the inspection procedure is not included, a file must be managed**. Such**

a file must carry at least a variable whose field is `PluVel`, according to `PluVelInspect` output (and `PluVelScalogram` input).

After the first step (or the direct jump to the second step), the function `DefPraramInteractive` is automatically called in order to allow the interactive choice of all the parameters required for the computations (Fig. 4.1). Moreover, a menu box allows the choice of the two options about the scalogram degeneration, i.e. use of both rainfall and velocity data and use of rainfall data only (Fig. 4.2.b).

All the functions described in Sections 4.3-4.8 are automatically called by `WADENOWGeneral`. Finally, the fact that such a script does not perform the operational use of the trained CNN should be noted.

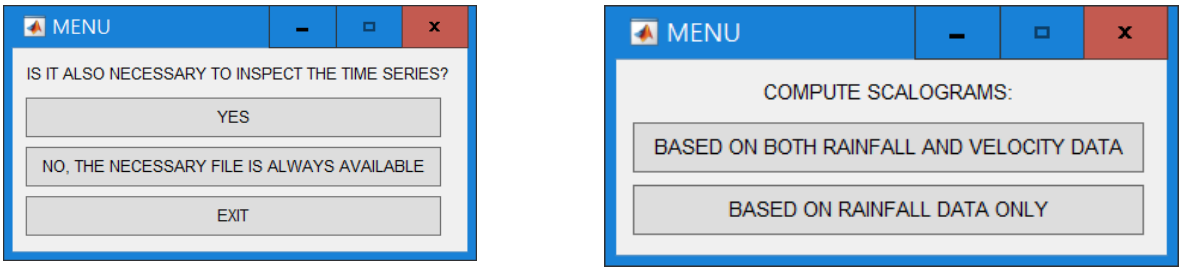

Figure 4.2 Main menu boxes of `WADENOWGeneral`.

4.3 Rainfall and velocity time series inspection

The function which allows the rainfall and velocity comparative inspection is **`PluVelInspect`**, The corresponding help is:

```
PluVel=PluVelInspect(MR1,MR7,MV)
PluVel=PluVelInspect(MR1,MR7,MV,d1,d2)
```

Let `MR1=[tR R1]` and `MR7=[tR R7]` be 1 d and 7 d cumulative rainfall time series and `MV=[tV V]` a velocity time series with sampling rate of 24 cpd (cycles per day).
This function allows the time series inspection in order to verify the nature of the velocity peaks. The rainfall and velocity time series are shown in two subplots and the user can zoom in the time series, leading to a segment with one of these criteria by acting on a popup (Fig. 4.3):

```
ZOOM ON 30 DAYS TIME SERIES AROUND A SELECTED TIME
ZOOM ON 2 MONTH TIME SERIES AROUND A SELECTED TIME
ZOOM ON ONE YEAR TIME SERIES AROUND A SELECTED TIME
FREE CHOICE OF TIME LIMITS
```

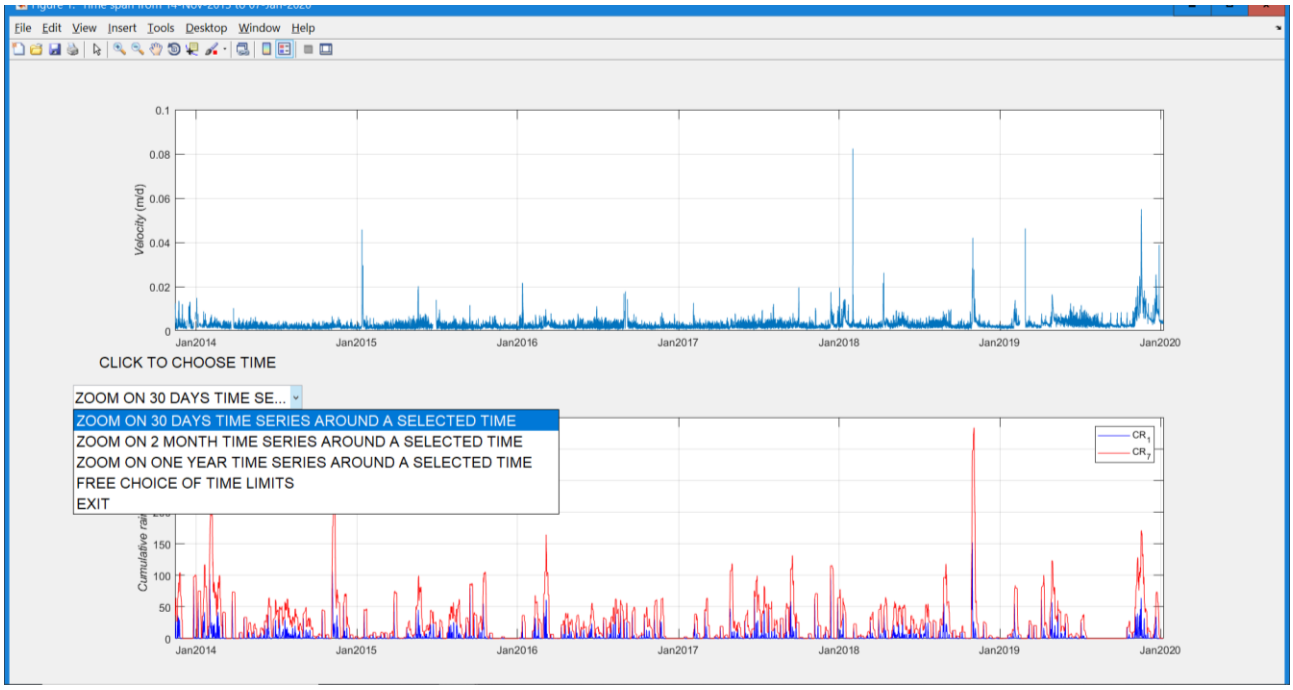

Figure 4.3 Complete common time series of velocities (upper panel) and 1d and 7d cumulative rainfall (lower panel) and menu for the choice of the action.

As a time series segment is selected, a figure with the zoom is shown and the user can verify if a possible velocity peak is real, i.e. rainfall triggered, or not. Moreover, a popup appears in the segment figure in order to allow the interactive removal of possible false peaks (Fig. 4.4).

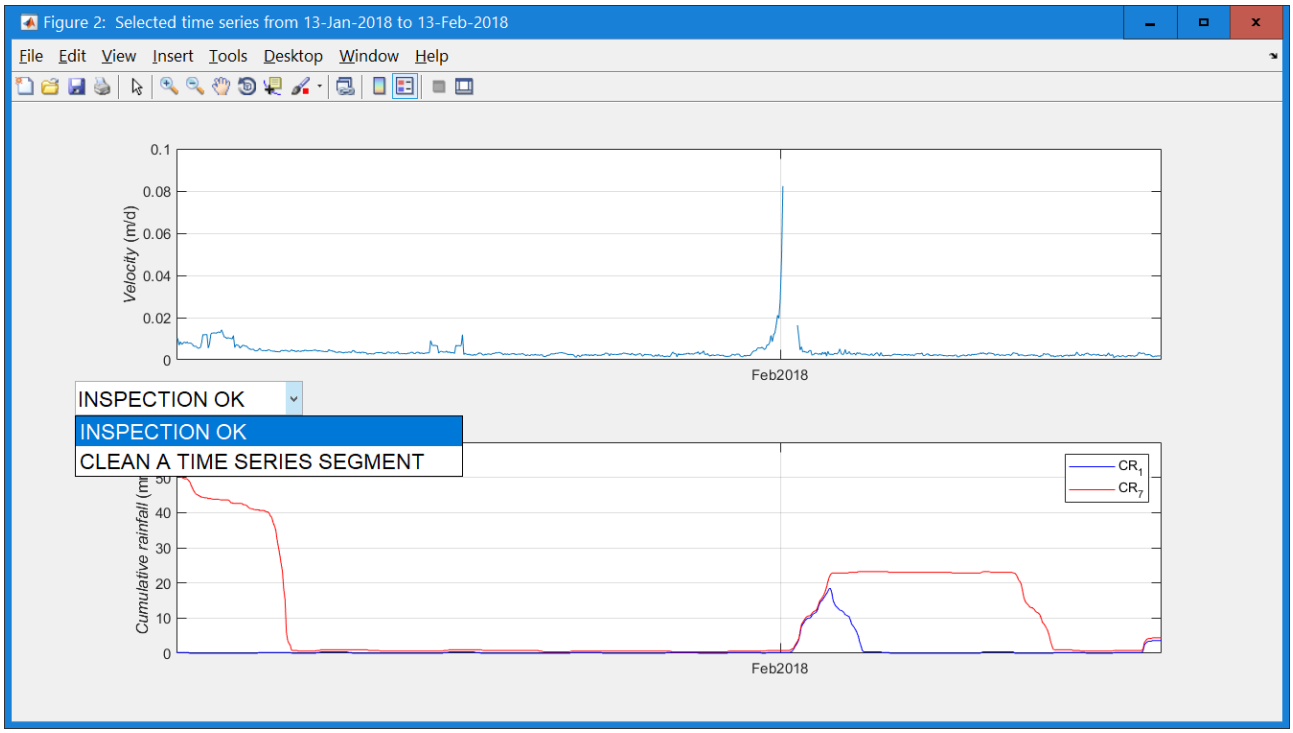

Figure 4.4 Selected segment of time series of velocities (upper panel) and 1d and 7d cumulative rainfall (lower panel) and menu for the choice of the action.

If d1 and d2 are defined and are MATLAB serial date numbers, they are the limits of the rainfall time series. If d1 and d2 are undefined or empty, the considered limits are max(tR(1),tV(1)) and min(tR(end),tV(end)) respectively.

The output matrix PluVel=[t R1 V] is generated according to the user's choices about the peak inspection and the management of time series limits. the general script for the Continuous Wavelet and Deep Learning based rainfall/velocity time series classification for nowcasting purposes.

If the user sees that an observed peak is false, he/she can choose the option CLEAN A TIME SERIES SEGMENT, interactively select the limits of this sub-segment and define a constant value of velocity in such a sub-segment.

4.4 Scalogram computation and image generation

The function that performs the scalogram computation and image generation is **PluVelScalogram**:

```
SigSegmData=PluVelScalogram(sigPluVel,ParamWN,chPlot)
```

Let sigPluVel be either a time series of rainfall and velocity data or a time series of rainfall data only. In particular,

- In the first case it is sigPluVel=[t Plu Vel], where Plu(k) is the pluviometric value (in mm) corresponding to t(k), time espressed in MATLAB serial number form, with t(k+1)-t(k)=1/24, i.e. 1h, for each k, and Vel(k) is the corresponding velocity (the sampling frequency is Fs=24). Another requirement is that t(1) and t(end) are integer numbers, i.e. are dates related to 0h.
- In the second case it is sigPluVel=[t Plu], with the same requirements about t and Plu.
- The elements of the input Plu can be differential or cumulative values (at some days or fraction of day), according to the user's preference.

This function computes the scalograms by means of Continuous Wavelet Transform (CWT). The parameters for CWT computation are taken from ParamWN, managed by means of DefParam or DefParamInteractive (if ParamWN is undefined or empty, DefParam is automatically called). For computation efficiency purposes, if amor is the selected wavelet, the CWT is computed by means of preliminary CWT filter bank generation.

If SigPluVel is a 3-columns matrix, each saved image (see below) shows two scalograms: the upper one is related to rainfall data, and the lower one is related to velocity data. If Nb>0 (see below), the velocity scalogram is zero-padded in order to align it to the rainfall scalogram (Fig. 4.5.a).
If SigPluVel is a 2-columns matrix, each output image shows a scalogram only, related to rainfall data (Fig. 4.5.b).
Each scalogram is computed for the reference date t(k) on the basis of these data taken from ParamWN:

`Na`: the starting date of the scalograms related to `t(k)` is `t(k)-Na`, where `Na` is expressed in days, under the condition `t(k)-Na>=t(1)` (if this condition is not fulfilled, the corresponding scalogram is not computed).

`Nb`: the ending date of the pluviometric scalogram related to `t(k)` is `t(k)+Nb`, where `Nb` is expressed in days, under the condition that `t(k)+Nb<=t(end)` (if such a condition is not fulfilled, the corresponding scalogram is not computed). The ending date of the velocity scalogram always is `t(k)`.

`NPerDay`: number of scalograms per day. The allowed values are 1 (one scalogram per day at 0h), 2 (two scalograms at 0h and 12 h), 3 (0h, 8h, 16h), 4 (0h, 6h, 12h, 18h), 6 (0h, 4h, 8h, 12h, 16h, 18h, 20h) 12 (even hours), 24.

`DateIn`: initial date for which the scalograms are computed (this date must be expressed as standard MATLAB serial number). The start time of the first computed scalogram is `max(d1-Na,t(1))`.

`DateFin`: final date for which the scalograms are computed (this date must be expressed as serial number). The end time of the last computed scalogram is `min(d2+Nb,t(end))`.

`ComPart`: common part of the output filename (folder name and constant part of the filename). Each filename is completed with the date string. For example, if `ComPart` is the string `pluvioData\LongBeach15days`, the filename of the scalogram of data related to the 15 d until the date 15 march 2020, 12:00:00 is the string `pluvioData\LongBeach15days_15-Mar-2020-120000.jpg`.

If `ParamWN` is a `char` variable instead of a `struct` one, it is supposed to be the name of a file having a field named `ParamWN`. If the file loading or the extraction of the `ParamWN` value is not successful, it is `ParamWN=[]`, also for the remaining cases of `ParamWN`.
If `ParamWN` is undefined or empty, a session of `DefParam` runs to generate `ParamWN`.

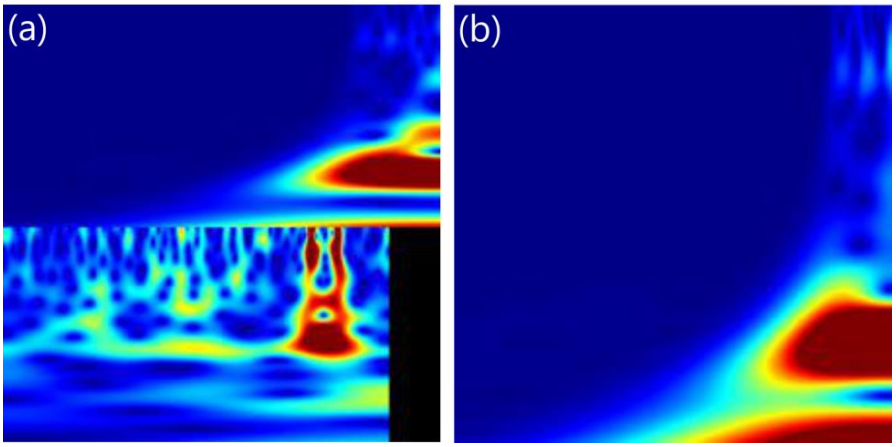

Figure 4.5. Examples of figures related to a same event in the case of velocity and rainfall scalograms (a) and rainfall scalogram only (b)

The remaining input argument is:

`chPlot`: vector of dates for which, besides the file generation, the scalogram are also shown in a two subplots figure, where the upper scalogram is the rainfall one and the lower

scalogram is the velocity one (if rainfall data only are used, a scalogram only is shown). The dates in `chPlot` must be coherent with the choice about `NPerDay`.

If `chPlot` is undefined or empty, no figures are shown.

The duration of each rainfall signal segment is `Na+Nb` days and the corresponding length is `(Na+Nb)*Fs+1`. The duration of each velocity signal segment is `Na` days, with `Na*Fs+1` length.

The output `SigSegmData` is a 4-column cell array such that `SigSegmData{k,1}` is the reference time of the k-th row, `SigSegmData{k,2}` is the initial time of the rainfall time series, `SigSegmData{k,3}` is the corresponding end time, `SigSegmData{k,4}` is the generated filename. In this way, some information about the signal segment which contributes to the k-th figure can be stored.

4.5 Velocity trend classification

The function that performs this step is **`trendClass`**:

```
[L,NL,T]=trendClass(sigIn,SigSegmData,ParamWN)
```

Output data:

`L`: N-by-4 matrix `L=[tRef,tIn,tFin,ClassTS]`, where `tRef(k)` is the reference time for the k-th velocity time series segment, `tIn=tRef-Nc`, `tFin=tRef+Nd` (see below) and `ClassTS` if the estimated class of this time series segment.

`NL`: number of possible input levels (7 or 4, see below).

`T`: table whose variables are the possible classifications (including 'undefined' at the first row; therefore, the table has either 8 or 5 rows), the corresponding outcomes and the percentages. The table is also shown on the MATLAB Command Window.

Input data:

`sigIn`: input signal. It can be either a two columns matrix `[t V]` or a three columns matrix `[t R V]`, where `t` is the time, `R` is the rainfall time series and `V` is the velocity time series.

`SigSegmData`: cell variable provided by `PluVelScalogram` to generate the scalogram-bases images for the CNN training. This variable is

```
SigSegmData={tRef,t1,t2,filena};
```

Moreover, these fields of `ParamWN` are used:

`Nc`: number of days to be considered before `tRef`. If it is empty, it is `Nc=5`;

`Nd`: number of days to be considered after `tRef`; If it is empty, it is `Nd=5`;

`Vmh`: scalar/vector/matrix for the management of thresholds for the segments classification. Options about such an argument:

- `Vmh` is a scalar. in this case, a 4-levels classification (including the transitions) is carried out, according to:

  `1: L, 2: L->H, 3: H, 4: H->L,`

  with the threshold `VH=Vmh`.

- `Vmh` is a 2-elements vector. In this case, a 7-levels classification is carried out with the thresholds `VM=Vmh(1)` and `VH=Vmh(2)` (the vector is sorted in ascending order). The levels `L`, `M` and `H` are characterized by the conditions `V<=M`, `M<V<=H` and `V>H` respectively. The outputs are:
  `1: L, 2: L->M, 3: M, 4: M->H (or also L->H),`
  `5: H, 6: H->M, 7: M->L (or also H->L).`
- `Vmh` is a 2-column matrix `Vmh=[t VH]`, where `VH(h)` is the time-dependent threshold at `t(h)`, as in the first case (four level output).
- `Vmh` is a 3-column matrix `Vmh=[t VM VH]`, where `VM(h)` and `VH(h)` are the time-dependent thresholds at `t(h)`, as in the second case.

If `ParamWN` is a `char` variable instead of a `struct` one, it is supposed to be the name of a file having a field named `ParamWN`. If the file loading or the extraction of the `ParamWN` value is not successful, it is `ParamWN=[]`, also for the remaining cases of `ParamWN`.
If `ParamWN` is undefined or empty, a session of `DefParam` runs to generate `ParamWN`.

An example of table provided by `trendClass` is

classification	number	percentage
'Undefined'	0	0
'L1'	1611	61.748
'L2'	199	7.6274
'L3'	456	17.478
'L4'	80	3.0663
'L5'	29	1.1115
'L6'	40	1.5332
'L7'	194	7.4358

The class `'Undefined'` corresponds to the set of time series segments that cannot be correctly classified in one of the classes L_1-L_7 (empty in the case of the shown table).

4.6 Data homogenization and augmentation

In order to have good results, the seven (or four) sets of input data should have a roughly equal size. Nevertheless, it is reasonable to expect that the L_1 cases are numerically prevalent over the others, and the extreme cases are probably few. For this reason, data augmentation could be required to have enough events L_4, L_5 and L_6. Data augmentation is a technique whose aim is the artificial creation of new training data from existing ones.

In general, data augmentation is carried out either by applying geometric transformations on the image, e.g. translations, rotations, cropping, flipping, scaling, shearing, or by addition data on the time series from which the image is generated (for a review, see e.g. Lashgari et al., 2020). Here, the used approach is the second one. Random variations are performed on the time series

related L_4, L_5 and L_6 by adding noise in order to have slightly varied scalograms and, therefore, varied input images. Since data augmentation adds variance without losing the information carried by the data, this allows both the reduction of the risk of overfitting and the improvement of the CNN accuracy on unseen data.

The function that performs this step is **`dataHomAug`**:

```
[IMDS,SLC,T]=dataHomAug(sigIn,SigSegmDataIn,LIn,TIn,ParamWN)
```

This function allows the homogeneization (in the sense of homogenization of cardinality of the corresponding sets) and, where necessary, the data augmentation for the time series of rainfall and, possibly, velocity by carrying out these operations:

- choice of amount of time series to be used for CNN training for each level. The choice is carried out by means of an input dialog box (Fig. 4.6);
- random selection of time series of the lower level, which typically is overrepresented;
- data augmentation for the time series of higher levels, which are underrepresented, including generation of scalogram-based images;
- transfer of available/generated images to the corresponding output folders;
- generation of the image datastores for the subsequent CNN training.

The computations are carried according to these input arguments:

`sigIn`: input time series, where it is either `sigIn=[t R V]` or `sigIn=[t V]`;

`SigSegmDataIn`: variable provided by a previous session of `PluVelScalogram`, where `SigSegmDataIn={tRef,t1,t2,filenaIn}`;

`LIn`, `TIn`: variable `LIn={tRef,tIn,tFin,ClassTS}` and table `Tin` provided by a previous session of `trendClass`;

Moreover, these fields from `ParamWN` are used:

`SigmaDA`: standard deviation of rainfall and velocity to be used in data augmentation. Options about `SigmaDA`:

- vector having two elements. In this case, it is `sigmaR=SigmaDA(1)` and `sigmaV=SigmaDA(2)`.
- Scalar. In this case, it is `sigmaR=SigmaDA`. Please note that `sigmaDA` must be coherent with the `sigIn`.
- Undefined or empty. In this case, `SigmaDA`, can be managed in an interactive way.

`FoldOut`: general folder of output data. For each level `Lk` (`k=1:4` or `k=1:7`, depending on the number of classes represented in `ClassTS`, last column vector of input matrix `LIn`, a nested folder whose name is `Lk` is generated and the corresponding output scalogram images are placed here. Clearly, possible undefined time series segments do not lead to images of the output datastore.

If `ParamWN` is a `char` variable instead of a `struct` one, it is supposed to be the name of a file having a field named `ParamWN`. If the file loading or the extraction of the `ParamWN` value is not successful, it is `ParamWN=[]`, also for the remaining cases of `ParamWN`.

If `ParamWN` is undefined or empty, a session of `DefParam` runs to generate `ParamWN`.

Output variables:

`IMDS`: output `image datastore` to be used for the CNN training;

`SLC`: cell array such that `SLC(h)` is the cell array

`SLC(h)={tRef,tIn,tFin,t1,t2,filena},`

where `tRef` is the reference time, `tIn` and `tFin` are the initial and final points of the times series segments used for trend classification, `t1` and `t2` are the initial and final times of the time series used for rainfall scalogram generation (if velocity data are also used, the corresponding initial and final point are `t1` and `tRef`), and `filena` are the filenames of the scalogram images placed in the output folders (in case of data augmentation, the generated files have suffix, before the extension, `"_m"`, where `m` is the progressive number of the copy of the time series segment to which random noise is added. It is `m>=1`);

`T`: table summarizing the results. For each level, the number of outcomes and the corresponding percentages are shown. Since this table is strictly related to the output data that lead to correctly classified images, transferred to the output datastore `IMDS`, the row `'Undefined'`, which is present in `TIn` (see also help `trendClass`), is not shown in `T` (examples of tables summarizing the statistics about the classified levels before and after data augmentation carried out with `dataHomAug` are shown in the next page).

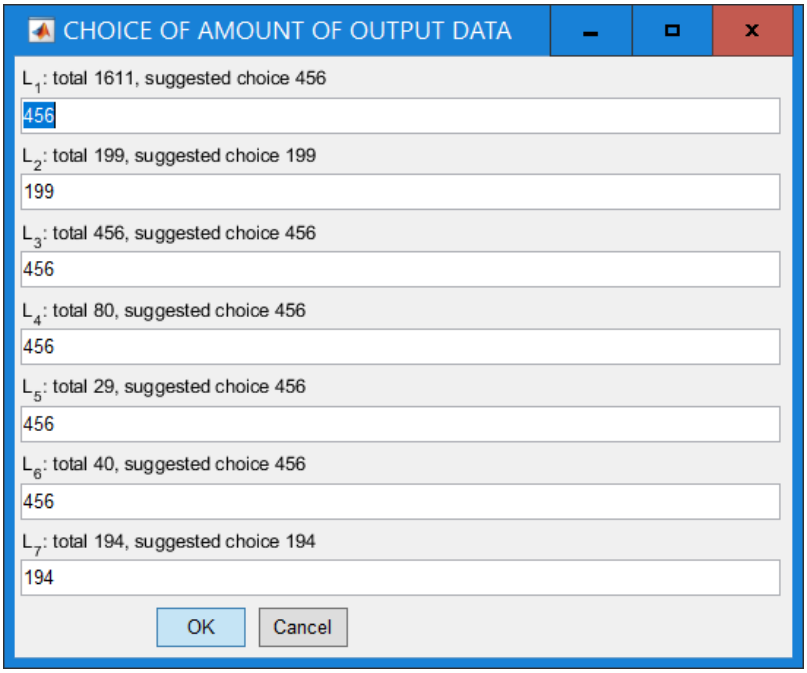

Figure 4.6. Input dialog box for the interactive choice of amount of output images for each class from L_1 to L_7.

Examples of tables summarizing the main data about time series classification before (left) and after (right) data homogeneization and augmentation:

Summary about time series classification:

classification	number	percentage
'Undefined'	0	0
'L1'	1611	61.748
'L2'	199	7.6274
'L3'	456	17.478
'L4'	80	3.0663
'L5'	29	1.1115
'L6'	40	1.5332
'L7'	194	7.4358

Summary after data augmentation:

classification	number	percentage
'L1'	456	18.088
'L2'	199	7.8937
'L3'	456	18.088
'L4'	375	14.875
'L5'	426	16.898
'L6'	415	16.462
'L7'	194	7.6954

4.7 Partition into training, validation and test datasets

The partition of the images datastore generated by using `dataHomAug` into training, validation and test datasets is carried out by means of **`dataTVT`** function:

```
[TrainDS,ValDS,TestDS]=dataTVT(IMDS,ParamWN,opRand)
```

This function provides the datasets `TrainDS`, `ValDS` and `TestDS` for training, validation and test on the basis of these input arguments:

`IMDS`: input dataset. If `IMDS` is undefined or empty, the name of the file with such data can be interactively managed. In this case, the file must contain a field whose name is `IMDS`;

`ParamWN.VectPart`: 3-elements vector such that `vectPart(1)` is the fraction of images for training, `vectPart(2)` is the fraction for validation and `vectPart(3)` is the fraction for test. The condition `sum(vectPart)<=1` must be satisfied. If `ParamWN` is a `char` variable, it is supposed to be the name of a file having a field named `ParamWN`. If the file loading or the extraction of the `ParamWN` value is not successful, it is `ParamWN=[]`. If `ParamWN` is undefined or empty, a session of `DefParam` runs to define `ParamWN`;

`opRand`: if `true`, the partition is carried out in random way. If `false`, the partition is with the initial order. If `opRand` is undefined or empty, the default value (`true`) is used.

4.8 Transfer learning or training resume

This step is the heart of the proposed procedure. The involved function is **`TransLearnPluVel`**.

```
net=TRLearnPluVel
net=TRLearnPluVel(ParamWN)
net=TRLearnPluVel(net)
net=TRLearnPluVel({net,ParamWN})
net=TRLearnPluVel([],TrainDS,ValDS,TestDS)
```

```
net=TRLearnPluVel(ParamWN,TrainDS,ValDS,TestDS)
net=TRLearnPluVel(net,TrainDS,ValDS,TestDS)
net=TRLearnPluVel({net,ParamWN},TrainDS,ValDS,TestDS)
```

This function allows the transfer learning of a CNN model or the training resume of a CNN. The trained CNN is the output variable `net`.
If the first argument is undefined or empty, i.e. in the cases

```
net=TRLearnPluVel
net=TRLearnPluVel([],TrainDS,ValDS,TestDS),
```

a menu box allows the choice between these options:

`TRANSFER LEARNING FROM A MODEL - DEFAULT PARAMETERS`

`TRANSFER LEARNING FROM A MODEL - MANAGE PARAMETERS FILE`

`RESUME CNN TRAINING - DEFAULT PARAMETERS`

`RESUME CNN TRAINING - MANAGE PARAMETERS FILE`

If the option `TRANSFER LEARNING FROM A MODEL - DEFAULT PARAMETERS` is chosen, the model and the other parameters carried by `ParamWN` generated from a session of `DefParam` are used (please note that a successful call of a model requires the corresponding support package, see Chapter 2).
If the option `TRANSFER LEARNING FROM A MODEL - MANAGE PARAMETERS FILE` is chosen, the model and the other parameters carried by `ParamWN` from a file interactively managed are used (**please note that the file of parameter data must have a field named `ParamWN`, case sensitive, and the requirement as above about the support package**).
If the option `RESUME CNN TRAINING - DEFAULT PARAMETERS` is chosen, a combo box allows the choice of the file with the CNN whose training should be resumed **Such a file must contain a field whose name is `net` (case sensitive)**. Moreover, the parameters carried by `ParamWN` generated by a session of `DefParam` are used.
If the option `RESUME CNN TRAINING - MANAGE PARAMETERS FILE` is chosen, two combo boxes allow the choices of the file with the CNN whose training should be (compulsory field: net) and of the parameters file (compulsory field: `ParamWN`)

If the first argument is the struct variable `ParamWN`, i.e. in the cases

```
net=TRLearnPluVel(ParamWN)
net=TRLearnPluVel(ParamWN,TrainDS,ValDS,TestDS),
```

the model to be used for the transfer learning is taken from `ParamWN`.
If the first argument is a CNN net, i.e. in the cases

```
net=TRLearnPluVel(net)
net=TRLearnPluVel(net,TrainDS,ValDS,TestDS),
```

this first argument is expected to be a CNN for the training resume.

If the first argument is a cell variable, i.e. in the cases

```
net=TRLearnPluVel({net,ParamWN})
```

`net=`**`TRLearnPluVel`**`({net,ParamWN},TrainDS,ValDS,TestDS),`
an element of this first argument is expected to be CNN for the training resume and the other argument is either a `ParamWN` variable or the name of a file having a field named `ParamWN`.

If the 2nd, 3rd and 4th input arguments are defined, they are assumed to be the training, validation and test image datasets respectively. The corresponding input options are

`net=`**`TRLearnPluVel`**`([],TrainDS,ValDS,TestDS)`
`net=`**`TRLearnPluVel`**`(ParamWN,TrainDS,ValDS,TestDS)`
`net=`**`TRLearnPluVel`**`(net,TrainDS,ValDS,TestDS).`

If one or more of these input arguments is undefined or empty, a combo box allows the choice of the corresponding file, which must carry at least the three fields `TrainDS`, `ValDS` and `TestDS` (case sensitive).

In the case of transfer learning, the necessary layer changes are automatically carried out taking into account the input image datasets.

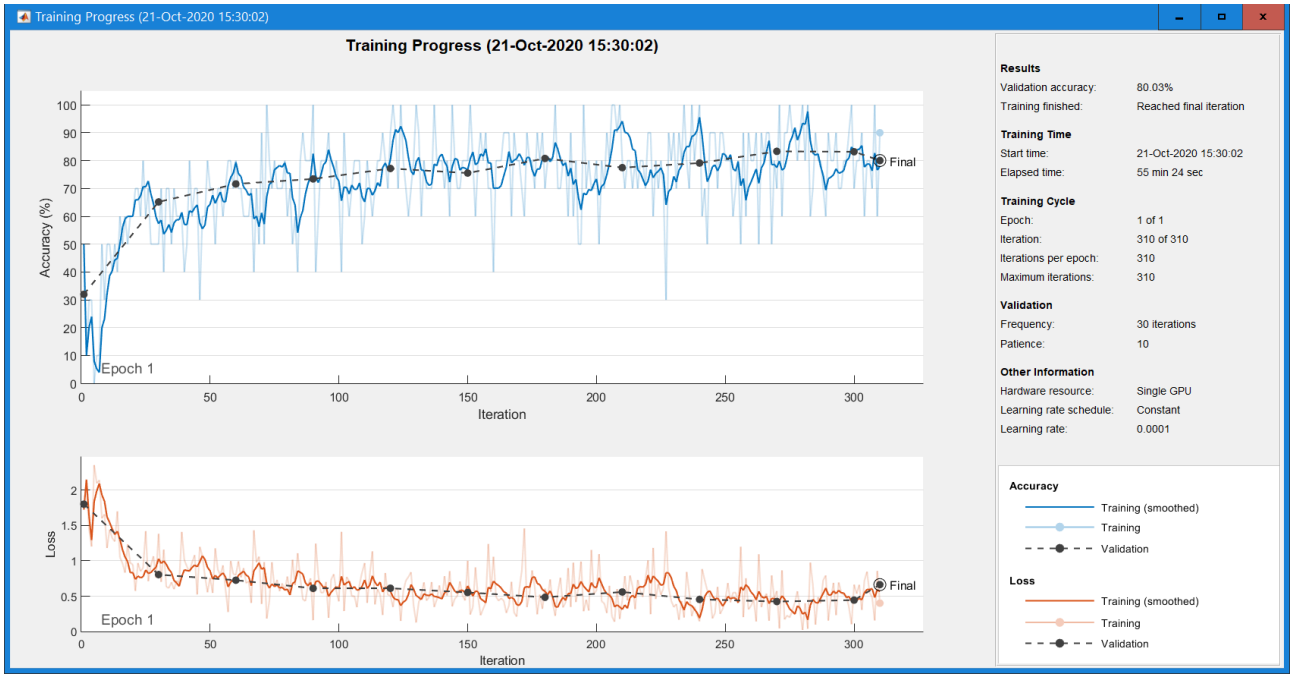

Figure 4.7 Example of plot of training progress

The fact that the script `WADENOWGeneral` is not suitable for training resume should be noted. The function `TRLearnPluVel` should directly be used instead, in general with the simplest configuration for the input arguments, i.e.

`net=`**`TRLearnPluVel`**

with the consequent loading of files already generated and saved in a previous session of the `WADENOW` toolbox, under the condition that the conditions above shown about the fields of the variables in these files are fulfilled.

Regardless to the fact that the process is a learning with new outputs of a pre-trained model or a training resume, the plots of training progress (Fig. 4.7) and confusion matrix (Fig. 4.8) ae shown.

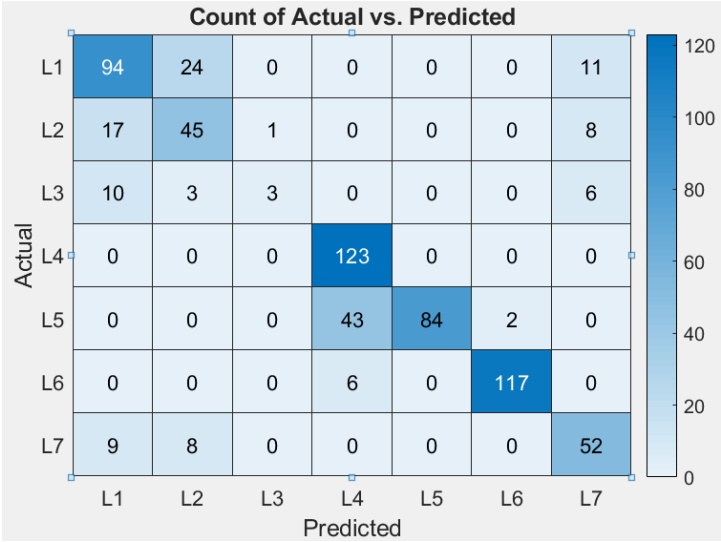

Figure 4.8 Example of confusion matrix

The confusion matrix shows the performance of a trained CNN of the test dataset. In the case shown in Fig. 4.8, among the 129 true L5 events, i.e. high velocity kinematics, 84 are correctly classified by the CNN as L_5, 43 are classified as transition mid/high velocity (L_4) and 2 are classified as transition high/mid velocity (L_6). Although the correct recognition is obtained for no more than 65% events, the results are very good because the alarm signal is emitted or maintained in 127 cases on 129 (98.5%).

4.9 Operational use of the trained CNN

The operational use of the trained CNN critically depends on the system which made available the rainfall and velocity data. The specific user's needs may require the study of a customized solution. However, there are some fixed points that need to be addressed by the user-developer and whose solution has to be implemented in the function or script for the CNN operation:

(1) execution of the operation script or function at scheduled times;
(2) access to a local or remote database where the data provided by the pluviometer and the position monitoring system are placed at other scheduled times;
(3) access, if applicable, to a database or to a web site with weather forecasts;
(4) processing of monitoring data aimed at providing rainfall and velocity time series compatible with the trained CNN (or CNNs if several points are to be surveyed and/or if spectrograms provided by rainfall data only are also considered);
(5) computation of the forecast and saving or the results to a specific file;
(6) if necessary, emission of an email and/or another signal for the decision maker.

It is important to underline that the MATLAB `startat` function, which allows the execution of a defined process at scheduled times, is not suitable in the case of a monitoring system. Instead, it is definitely preferable to use an approach that directly involves the operating system (OS) and performs the following periodic operations:

(a) MATLAB start at scheduled times using the specific OS task scheduler (see below);

(b) start of the MATLAB script that performs the actual calculations through the startup file (see below);

(c) MATLAB close as the operations are completed.

In the case of Windows10, a step-by-step tutorial about the Task Scheduler can be found in https://www.windowscentral.com/how-create-automated-task-using-task-scheduler-windows-10. In the case of MacOS, the task scheduling can be managed with the approach described e.g. in https://www.addictivetips.com/mac-os/how-to-schedule-tasks-on-macos/. In the case of Unix-like OSs (Unix, Linux and similar, the utility `cron` can be used, as depicted in https://opensource.com/article/17/11/how-use-cron-linux. Regardless to the specific OS, the scheduler is used to wake up MATLAB.

The MATLAB script developed in the case of Perarolo di Cadore landslide (Teza et al., 2020) is **`PluVelOpera`**. It performs all the operations (I)-(VI) above described. It is a script and not a function because all the variables are managed from files (no variables are taken from the MCW) and all the results are saved on files and/or sent by email. On the other part, MATLAB is automatically closed as the **`PluVelOpera`** execution is completed.

In order to execute **`PluVelOpera`** (or a specific user-defined script), a `startup` file is necessary. As shown in Subsection 2.3, such a file also allows to add the path of the folder where `WADENOW` is placed. With the same example shown in Subsection 2.3, if the user want to add the search path by means of `addpath`, the command lines to be written on `startup.m` file are

```
addpath C:\Users\HJ\Documents\MATLAB\WADENOW;     % addpath row
PluVelOpera                                       % execute PluVelOpera
```

If the user want to change the folder to the `WADENOW` one, the possible commands are, for example,

```
dirWN='C:\Users\HJ\Documents\MATLAB\WADENOW';
cd(dirWN);
PluVelOpera;
```

(`addpath` in this case is not strictly necessary because the folder is changed. Anyway, if `addpath` is carried out by means of `Set Path`, the corresponding row is unnecessary. Please also note that the operating systems Unix, Linux and MacOS require '/' instead of '\'). Please recall that `startup.m` must be placed in the MATLAB directory (for example, `C:\Program Files\MATLAB\R2018a\bin` in the case of R2018a in Windows).

As MATLAB starts at the time scheduled by acting on the OS, the `startup` file runs and **`PluVelOpera`** is executed. This script performs there operations:

1) Access via `FTP` to a remote database where the data provided by pluviometer and total station are automatically placed as ASCII files by the monitoring times at scheduled times. For example, the first ten rows of the file provided by the RTS operating at Perarolo are

```
14/11/2013 13:00    R5    235.16553    99.13675    177.393     -94.1715    -151.484    2.4131
14/11/2013 13:01    P23   289.24806    77.43665    233.3251    -216.9228   -36.9491    81.1263
14/11/2013 13:01    P13   291.55386    77.64654    201.8308    -189.0034   -25.2417    69.5501
14/11/2013 13:02    P2    292.5236     77.56857    235.1468    -220.3893   -26.0038    81.3018
14/11/2013 13:03    R3    293.28523    72.04332    407.1313    -367.9338   -38.8272    173.4206
14/11/2013 13:03    P3    294.01576    78.33493    212.9589    -201.0279   -18.9921    71.2145
14/11/2013 13:04    P17   294.41126    77.65674    254.4467    -239.254    -21.067     87.6435
14/11/2013 13:04    P1    296.13365    78.40234    240.4847    -227.5707   -13.8805    80.179
14/11/2013 13:05    P14   296.57255    79.13128    193.5289    -184.0883   -10.0089    62.4258
14/11/2013 13:05    P21   296.65589    83.76059    166.571 -   162.0484    -8.6298     42.1099
```

where the Cartesian coordinates of each reflector are the last three columns. Two sample files are added to WADENOW toolbox for tutorial purposes (`OperationPluvio.txt` and `OperationST.txt`)

2) Access via FTP to a remote database with weather forecasts (there forecasts typically are provided at scheduled times by an external source, ARPAV DOLOMITI Meteo in the case of Perarolo di Cadore landslide);

3) Processing of monitoring data to provide rainfall and velocity time series compatible with the trained CNN or CNNs. The rainfall time series is obtained by means of `extraPluvio` and `cumPuvio` functions, described in Subsection 3.2.1. The velocity time series is provided by `extraRTS2array` function (please see Subsection 3.2.2), which extracts the data related to the last month and provides a coordinate array, and `velArray` function (see 3.2.3), which computes the velocities.

4) Computation of the forecasts by calling the ancillary function `OutLevel` which, in turn, calls `PluVelScalogram` to generate the scalogram file and the MATLAB built-in function classify to evaluate the output level. The data are saved as a row

```
[tc level],
```

where `tc` is the computation time and `level` is the foreseen level, appended to a specific output file. If the RTS correctly sends its data, the CNN related to both rainfall and velocity time series is used to provide the output. If no RTS data are available, the second CNN, trained with rainfall data only, is used to evaluate the output. Moreover, the forecast is compared with the cumulative rainfall and the velocity. In particular, if the one-day cumulative rainfall CR_1 reaches `10*SigmaDA(1)` and/or the mean velocity is higher than `Vmh(2)` (`max(Vmh(:,3))` if the velocity thresholds are time-dependent) for at least two times, and a result L_4, L_2 or L_3 is found, a warning message is generated (`SigmaDA` and `Vmh` are among the `ParamWN` fields, see Subsection 4.1 about their definition and their admitted values).

5) Emission of an email to one or more destinations, with the relevant information for the decision maker. The email text is, depending on the result, one of the following:

```
Time yyyy-mm-dd - LOW VELOCITY
Time yyyy-mm-dd - PRE-ALERT, TRANSITION LOW/MEDIUM VELOCITY
Time yyyy-mm-dd - ALWAYS PRE-ALERT, MEDIUM VELOCITY
Time yyyy-mm-dd - ALARM, TRANSITION MEDIUM/HIGH VELOCITY
Time yyyy-mm-dd - ALWAYS ALARM, HIGH VELOCITY
Time yyyy-mm-dd - TOWARDS ALARM RETURN, TRANSITION FROM HIGH
  TO MEDIUM VELOCITY
Time yyyy-mm-dd - TOWARDS PRE-ALERT RETURN, TRANSITION FROM
  HIGH TO MEDIUM VELOCITY
```

If only rainfall data are used to provide forecasts because of unavailability of velocity data, the possible messages are:

```
Time yyyy-mm-dd - LOW VELOCITY (FROM RAINFALL ONLY)
```

Time yyyy-mm-dd - PRE-ALERT, TRANSITION LOW/MEDIUM VELOCITY (FROM RAINFALL ONLY)

Time yyyy-mm-dd - ALWAYS PRE-ALERT, MEDIUM VELOCITY (FROM RAINFALL ONLY)

Time yyyy-mm-dd - ALARM, TRANSITION MEDIUM/HIGH VELOCITY (FROM RAINFALL ONLY)

Time yyyy-mm-dd - ALWAYS ALARM, HIGH VELOCITY (FROM RAINFALL ONLY)

Time yyyy-mm-dd - TOWARDS ALARM RETURN, TRANSITION FROM HIGH TO MEDIUM VELOCITY (FROM RAINFALL ONLY)

Time yyyy-mm-dd - TOWARDS PRE-ALERT RETURN, TRANSITION FROM HIGH TO MEDIUM VELOCITY (FROM RAINFALL ONLY)

If a result L_1, L_2 or L_3 is obtained, but it is not coherent with the rainfall and velocity time series, one of these message is sent:

Time yyyy-mm-dd – WARNING - LOW VELOCITY BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Time yyyy-mm-dd – WARNING – PRE-ALERT BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Time yyyy-mm-dd – WARNING – ALWAYS PRE-ALERT BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Time yyyy-mm-dd – WARNING - LOW VELOCITY (FROM RAINFALL ONLY) BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Time yyyy-mm-dd – WARNING – PRE-ALERT (FROM RAINFALL ONLY) BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Time yyyy-mm-dd – WARNING – ALWAYS PRE-ALERT (FROM RAINFALL ONLY) BUT THE FORECAST IS NOT COHERENT WITH THE RAINFALL/VELOCITY DISTRIBUTION

Finally, if neither rainfall-velocity based nor rainfall only forecasts can be provided, this message is sent:

6) Time yyyy-mm-dd – WARNING - THE CNN-BASED SYSTEM IS UNABLE TO PROVIDE FORECASTS

7) MATLAB exit as the computations and data saving are completed.

In order to make easier the management of the filenames, folders, FTP site and email addresses, an Excel `.xls` file is used. This file is named `OperaPreferences.xlsx`. The data stored in `OperaPreferences.xlsx` are accessed by means of the ancillary function `readFOP`, called by **`PluVelOpera`**.

An example of `OperaPreferences.xlsx`:

	A	B	C	D
1	**Element**	**Variable in PluVelOpera**	**Value**	**Note**
2	ParamWN file	ParamWNfile	ParamWN.mat	
3	Folder of output data - Rainfall & velocity	FoldOperaRV	OutputRV	
4	Common part scalogram file - Rainfall & velocity	FileSGRV	ScalogramRV	
5	CNN file -Rainfall & velocity	NetFinalRV	NetFinalRV	
6	Results output file - Rainfall & velocity	OutpuFileRV	LevelRV	
7	Folder of output data - Rainfall only	FoldOperaR	OuputR	
8	Common part scalogram file - Rainfall only	FileSGR	ScalogramR	
9	Results output file - Rainfall only	OuputFileR	LevelR	
10	CNN file -Rainfall only	NetFinalR	NetFinalR	
11	FTP site	ftpSite	frane.dicea.unipd.it	
12	FTP user	ftpUser	xxxxxx	
13	FTP password	ftpPwd	xxxxxx	
14	FTP folder	dirTRfiles	PeraTopoRain	
15	FTP System	ftpSystem	unix	unix or windows
16	FTP rainfall file (folder excluded)	PluFilena	Pera_pluvio.txt	
17	FTP position file (folder excluded)	PosFilena	Misure_ST.txt	
18	FTP file weather forecast (folder excluded)	WeatherFilena	Forecast_NOD.txt	
19	emailAddress (email source)	emailAddress	name.familyname@gmail.com	
20	SMTP server	SMTP_Server	smtp.gmail.com	
21	SMTP username	SMTP_Username	name.familyname	
22	SMTP Password	SMTP_Password	xxxxxx	
23	Email subject	emailSubject	Perarolo Monitoring	
24	EmailAddress (receiver(s))	emailReceivers	receiver1@gmail.com,receiver2@gmail.com	
25				

All the necessary functions are called in an entirely automatic way and the access to the site FTP is entirely automatic. The CNN operation dos not require any action by the user.

Finally, since the MATLAB `exit` command is included at the last row of the **`PluVelOpera`** script, the user should comment such a command in the phase of customizing/debugging. Such a command will be uncommented as **`PluVelOpera`** is operationally used. Moreover, the row

```
PluVelOpera
```

should be added to the startup file (see above) not before the debugging is complete and the system enters in operational use.

5. List of developed MATLAB functions

For more information about the main functions and scripts, please see their descriptions in Chapter 3 and 4 as well as their helps. For example, to see the help of `PluVelScalogram` function, please type

```
help PluVelScalogram
```

on the MCW. The helps of ancillary functions are also available.

5.1 Main functions and scripts

All files listed here, with the exception of `OperaPreferences.xlsx`, are `.m` files (functions or scripts)

Note: WN: main component of `WADENOW` toolbox.

P: function for the data preparation.

`cell2posArray` (P, described in 3.2.1)

`cumPluvio` (P, described in 3.1)

`dataHomAug` (WN, described in 4.6)

`dataTVT` (WN, described in 4.7)

`DefParam` (WN, **function for the parameter definition**, described in 4.1)

`DefParamInteractive` (WN, **function for the parameter definition**, described in 4.1)

`extraPluvio` (P, described in 3.1)

`extraRTS2array` (P/WN, described in 3.2.2)

`extraTarget2cell` (P, described in 3.2.1)

`OperaPreferences.xlsx` (WN, Excel file for the management of data for CNN operation, described in 4.9)

`PluVelExample` (P, tutorial file described in 3.2.3)

`PluVelInspect` (WN, described in 4.3)

`PluVelOpera` (WN, described in 4.9)

`PluVelScalogram` (WN, described in 4.4)

`SelectMeanVelocity` (P, described in 3.2.3)

`TRlearnPluVel` (WN, described in 4.8)

`TrendClass` (WN, described in 4.5)

`velArray` (P, described in 3.2.3)

`WADENOWGeneral` (WN, described in 4.2)

5.2 Ancillary functions

These functions are called from the main functions. However, they could also be used in an independent way. There are not described in this User's Guide, but their `help` are available.

`OutLevel`

`plotRV`

`plotScalograms`

`readFOP`

`TimeSeriesLimCheck`

5.3 External MATLAB functions

In order to allow the choice of a general analytic Morlet wavelet, modified versions of these functions by Erickson (2020) are added to `WADENOW`:

`contwt`

`wave_bases`

These functions are modified versions of the ones provided by Torrance and Compo (1998). Further modification were carried out by G. Teza, in accordance with Mallat (2009), to allow the choice of σ_t (Erickson, 2020 allows the choice of ω_0 only).

5.4 Sample data for tutorial purposes

These Excel files are added in the folder `sampleFiles`, subfolder of `WADENOW` folder, as samples of data for training:

`Meteo_060120.xlsx` (rainfall data)

`Prisma_P1.xlsx` (position data)

`Prisma_P4.xlsx` (position data)

`Prisma_P5.xlsx` (position data)

`Prisma_P6.xlsx` (position data)

`Prisma_R1.xlsx` (position data)

These ASCII sample files carry data for CNN operation:

`OperationPluvio.txt` (rainfall data)

`OperationST.txt` (position data=

6. References

Addison, P.S., 2020. The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance, 2nd Edition, Boca Raton FL: CRC Press, 472 pp.

Cohen, M.X., 2019. A better way to define and describe Morlet wavelets for time-frequency analysis. NeuroImage, 199:81-86, doi: 10.1016/j.neuroimage.2019.05.048.

Erickson, J., 2020. Continuous wavelet transform and inverse (MATLAB Central File Exchange). Available online at: https://www.mathworks.com/matlabcentral/fileexchange/20821-continuous-wavelet-transform-and-inverse (accessed: 29/10/2020).

Goupillaud, P., Grossman, A., Morlet, J., 1984. Cycle-Octave and Related Transforms in Seismic Signal Analysis. Geoexploration, 23:85-102.

Lashgari, E., Dehua Liang, D., Maoz, U., 2020. Data augmentation for deep-learning-based electroencephalography. Journal of Neuroscience Methods, 346, 108885:1-26, doi: 10.1016/j.jneumeth.2020.108885.

Lilly, J.M., 2017. Element analysis: a wavelet-based method for analysing time-localized events in noisy time series. Proceedings of the Royal Society A, 473: 20160776, 2017, 1–28, doi: 10.1098/rspa.2016.0776.

Mallat, S., 2009. A Wavelet tour of signal processing, 3rd Edition, Burlington, MA: Academic Press, 832 pp.

Simonyan, K., Zisserman, A., 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition, in: Proc. of 3rd International Conference on Learning Representations (ICLR2015), San Diego CA, 7-9 May 2015. https://arxiv.org/abs/1409.1556.

Teza, G., Cola, S., Brezzi, L., Galgaro, A., 2022. WADENOW: A MATLAB toolbox for early forecasting of the velocity trend of a rainfall-triggered landslide by means of continuous wavelet transform and deep learning. Geosciences, 12(5), 205. https://doi.org/10.3390/geosciences12050205.

Torrence, C., Compo, G.P., 1998. A Practical Guide to Wavelet Analysis. Bulletin of the American Meteorological Society, 79(1):61-78, doi: 10.1175/1520-0477(1998)079<3C0061:APGTWA>3E2.0.CO;2.

Weiss, K., Khoshgoftaar, T.M., 2016. A survey of transfer learning, Journal of Big Data, 3(1):1-40, doi: 10.1186/s40537-016-0043-6.