**M³Fusion: A Deep Learning Architecture for Multiscale Multimodal Multitemporal Satellite Data Fusion**

# $M^3Fusion$: A Deep Learning Architecture for Multi-{Scale/Modal/Temporal} satellite data fusion

P. Benedetti, D. Ienco, R. Gaetano, K. Ose, R.G. Pensa and S. Dupuy

***Abstract*—Modern Earth Observation systems provide remote sensing data at different temporal and spatial resolutions. Among all the available spatial mission, today the Sentinel-2 program supplies high temporal (every 5 days) and high spatial resolution (10m) images that can be useful to monitor land cover dynamics. On the other hand, Very High Spatial Resolution (VHSR) imagery is still essential to figure out land cover mapping characterized by fine spatial patterns. Understanding how to jointly leverage these complementary sources in an efficient way when dealing with land cover mapping is a current challenge in remote sensing. With the aim of providing land cover mapping through the fusion of multi-temporal High Spatial Resolution and VHSR satellite images, we propose a suitable end-to-end Deep Learning framework, namely $M^3Fusion$, which is able to simultaneously leverage the temporal knowledge contained in time series data as well as the fine spatial information available in VHSR images. Experiments carried out on the *Reunion Island* study area confirm the quality of our proposal considering both quantitative and qualitative aspects.**

***Index Terms*—Land Cover Mapping, Data Fusion, Deep Learning, Satellite Image Time series, Very High Spatial Resolution, Sentinel-2.**

## I. Introduction

Modern Earth Observation (EO) systems produce huge volumes of data every day. Earth Observation programs (e.g., Copernicus) supply image acquisition at high spatial resolution (10m) with high temporal revisit period (every 5 days). This information can be organized into time series of high-resolution satellite imagery (SITS) that are particular useful for area monitoring over time. Other Earth Observation programs are able to provide image information at finer spatial resolution (between 0.5 to 2m) but with a low revisiting frequency. An example of EO program that supplies this kind of information is the SPOT6/7 mission that produces images with a spatial resolution of 1.5m. Such kind of images supply Very High Spatial Resolution (VHRS) information and they are extremely useful to characterize land use or land cover by means of their spatial structure [1].

P. Benedetti, D. Ienco and K. Ose are with UMR-TETIS laboratory, IRSTEA, University of Montpellier, Montpellier, France (email: dino.ienco@irstea.fr; paola.benedetti@irstea.fr; kenji.ose@irstea.fr).

D. Ienco is with LIRMM laboratory, Montpellier, France.

R. Gaetano is with CIRAD, UMR TETIS, 500 Rue J.-F. Breton, F-34000 Montpellier, France and with UMR TETIS, Univ. Montpellier, AgroParisTech, CIRAD, CNRS, IRSTEA, Montpellier, France (email: raffaele.gaetano@cirad.fr).

P. Benedetti and R. G. Pensa are with the University of Turin, Computer Science Department, I-10149, Turin, Italy (email: ruggero.pensa@unito.it).

S. Dupuy is with CIRAD, UMR TETIS, F-97410 Saint-Pierre, Runion, France and UMR TETIS, Univ. Montpellier, AgroParisTech, CIRAD, CNRS, IRSTEA, Montpellier, France (email: stephane.dupuy@cirad.fr).

In the context of land use and land cover classification, employing high spatial resolution (HSR) time series, instead of a single image of the same resolution, can be useful to distinguish land usage classes according to their temporal profile or evolution [2]. On the other hand, the use of fine spatial information (VHSR images) helps to differentiate other kind of classes that need spatial context information at a finer scale [3].

Due to the diverse, although complementary, information carried out by each of these different Earth Observation sources, how to intelligently combine satellite image time series and VHSR images via a dedicated fusion process, for a particular task at hand, constitutes an important challenge in the field of remote sensing [4], [3].

Considering data fusion at sensor level [3], several works exist that combine time series of satellite images at different resolutions together. For instance, in [5], the authors propose two methods to combine MODIS and LANDSAT time series images to produce a synthetic daily surface reflectance product at ETM+ spatial resolution. [6] extends this work providing a method that deal with cloudy phenomena as well as scales up over big surfaces. In [7], the authors propose a novel approach to combine two VHSR images (acquired at two different timestamps on the same area) coming from different sensors. Also in this case, the fusion process produces new synthetic surface reflectance images.

Conversely, when a particular task needs to be addressed, a different data fusion scenario is considered (fusion at feature level [3]). For instance, when multiple sources of remote sensing data are combined to deal with land cover/land use mapping, the results of the fusion process are not new synthetic images but directly the land cover classification. For instance, [8], [9] do not produce reflectance product but they directly solve the particular task at hand [3]. In both research studies, they first extract an independent set of features for each data source (time series, VHSR image) and, successively, they stack these features together to feed a traditional supervised learning method (i. e., Random Forest).

Recently, the deep learning revolution [10] has shown that neural network models are well adapted tools for automatically managing and classifying remote sensing data. The main characteristic of this type of model is the ability to simultaneously extract features optimized for image classification and the associated classifier. This advantage is fundamental in a data fusion process such as the one involving high resolution time series (i. e. Sentinel-2) and VHSR data (i. e. Spot6/7). Recent works have demonstrated the quality of deep learning for remote sensing data fusion. [11] introduces a regression deep

learning architecture to infer NDVI information for cloudy areas exploiting information coming from Sentinel 1 and Sentinel 2 times series available before and after the date affected by the cloud phenomena. In [12] the authors propose a deep learning architecture to fuse together hyperspectral and LIDAR signals with the goal to produce a land cover map. [13] proposes to exploit deep learning to combine PAN and MS information still to cope with land cover classification.

To the best of our knowledge, no Deep Learning architecture has already been introduced to deal with the challenging fusion problem involving optical High-Resolution Satellite Image Time Series and Very High Resolution imagery to provide land cover mapping [14].

As regards deep learning methods, we distinguish two main families of approaches: convolutional neural networks [10] (CNN) and recurrent neural networks [15] (RNN). CNN are well suited to model the spatial autocorrelation available in an image. RNN networks, instead, are specifically tailored to manage time dependencies [16], [17], [18] from multidimensional time series. In this article, we propose to leverage both CNN and RNN to address the fusion problem between a HSR time series of Sentinel-2 images and a single VHSR scene (SPOT6/7) on the same study area with the goal of performing land use mapping.

The method we propose, named $M^3Fusion$ (Multi-Scale/Modal/Temporal Fusion), is a deep learning architecture that integrates both a CNN module (to integrate VHSR information) and an RNN module (to manage HSR time series information) in an end-to-end learning process. Differently from general data fusion approaches [5], [6], [7], [3] in which the result is a set of new synthetic surface reflectance images, the outcome of our deep-learning based data fusion process is the final land cover classification avoiding the generation of any other intermediate product. Each information source is integrated through its dedicated module and the extracted descriptors are then concatenated to perform the final classification. All the non-linear transformations are learned together resulting in an architecture that is able to manage, simultaneously, multi-temporal and multi-scale information, thus enabling the extraction of complementary and diversely useful features for land use mapping.

To validate our approach, we conduct experiments on a data set regarding the *Reunion Island* site. This site is a French Overseas Department located in the Indian Ocean (east of Madagascar) and it will be described in Section II. The rest of the article is organized as follows: Section III introduces the $M^3Fusion$ Deep Learning Architecture for the multi-source classification process. The experimental setting and the findings are discussed in Section IV. Finally, conclusions are drawn in Section V.

## II. Data

The study was carried out on the Reunion Island, a French overseas department located in the Indian Ocean. The dataset consists of a time series of 34 Sentinel-2 (S2) images acquired between April 2016 and May 2017, as well as a very high spatial resolution (VHSR) SPOT6/7 image acquired in April 2016 and covering the whole island. The S2 images used are those provided at level 2A by the THEIA pole [1], where the bands at 20m resolution were resampled at 10m via bicubic interpolation. A preprocessing was performed to fill cloudy observations through a linear multi-temporal interpolation over each band (cfr. *Temporal Gapfilling*, [8]), and six radiometric indices were calculated for each date: NDVI, NDWI, brightness index (BI), NDVI and NDWI of infrared means (MNDVI and MNDWI), and vegetation index Red-Edge (RNDVI) [8], [9]. A total of 16 variables (10 surface reflectances plus 6 indices) are considered for each pixel of each image in the time series.

The SPOT6/7 image, acquired on April 6th 2016 and originally consisting of a 1.5 m panchromatic band and 4 multispectral bands (blue, green, red and near infrared) at 6 m resolution, was pansharpened to produce a single multispectral image at 1.5 m resolution and then resampled at 2 m via bicubic interpolation because of the network architecture learning requirements[2]. Its final size is 33 280 $\times$ 29 565 pixels on 5 bands (4 *Top of Atmosphere* reflectance plus the NDVI). This image was also used as a reference to realign the different images in the time series by searching and mapping anchor points, in order to improve the spatial coherence between the different sources.

The field database was built from various sources: (i) the *Registre parcellaire graphique* (RPG) reference data of 2014, (ii) GPS records from June 2017 and (iii) photo interpretation of the VHSR image conducted by an expert, with knowledge of the territory, for distinguishing between natural and urban spaces. RPG is part of the European Land Parcel Identification System (LPIS), provided by the French Agency for services and payment. The RPG supplies a thematic layer (in vector format) with information about the land cover for each polygon (vector) it contains. All polygon contours have been resumed using the VHSR image as a reference. The final dataset is composed of a total of 322 748 pixels (2 656 objects) distributed over 13 classes, as indicated in Table I.

| **Class** | Label | **# Objects** | **# Pixels** |
|---|---|---|---|
| 1 | *Crop Cultivations* | 380 | 12090 |
| 2 | *Sugar cane* | 496 | 84136 |
| 3 | *Orchards* | 299 | 15477 |
| 4 | *Forest plantations* | 67 | 9783 |
| 5 | *Meadow* | 257 | 50596 |
| 6 | *Forest* | 292 | 55108 |
| 7 | *Shrubby savannah* | 371 | 20287 |
| 8 | *Herbaceous savannah* | 78 | 5978 |
| 9 | *Bare rocks* | 107 | 18659 |
| 10 | *Urbanized areas* | 125 | 36178 |
| 11 | *Greenhouse crops* | 50 | 1877 |
| 12 | *Water Surfaces* | 96 | 7349 |
| 13 | *Shadows* | 38 | 5230 |

Table I: Characteristics of the Reunion Dataset

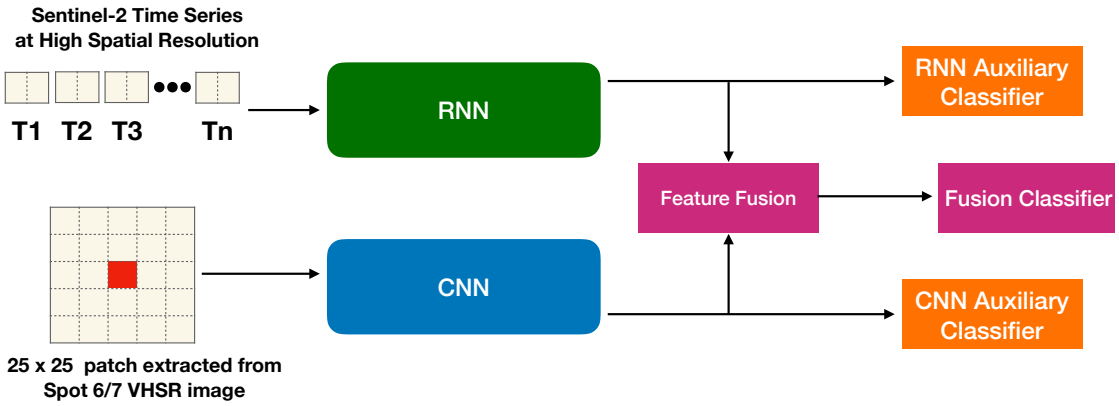



Figure 1: Visual representation of $M^3Fusion$.

## III. Contributions

### A. $M^3Fusion$ model overview

Figure 1 visually describes the Multi-Scale/Modal/Temporal Fusion ($M^3Fusion$) approach proposed in this work. First of all, we define the input data for our deep learning model. $M^3Fusion$ takes as input a dataset $\{(x_i, y_i)\}_{i=1}^{M}$ where each example is associated with a class value $y_i \in 1, ..., C$. An example $x_i$ is defined as a pair $x_i = (ts_i, patch_i)$ such that $ts_i$ is the (multidimensional) time series of a Sentinel-2 pixel (10 m resolution) and $patch_i$ is a subset of the image Spot6/7 (at 2 m resolution) centered around the corresponding Sentinel-2 pixel. Note here that every exemple is purposely built to encompass two different acquisition modes at two different scales for a given sample area: a pixel-based spectral dynamic via multi-temporal HSR data, and a patch-based fine-scale spatial/contextual information via the single date VHSR scene. For every $patch_i$, we fix the window size to 25×25 pixels on the Spot6/7 (which corresponds to a window size 5x5 on a Sentinel-2 image) centered around a Sentinel-2 pixel described by the corresponding $ts_i$.

In order to merge the temporal information (Sentinel-2) and the VHSR information (Spot 6/7), we designed a deep learning architecture which has two parallel branches, one for each of the two modes (spatial/temporal). For the Sentinel-2 pixel-based time series we use a Recurrent Neural Network (RNN) architecture. In particular, we used a Gated Recurrent Unit (GRU) introduced in [20] which has already demonstrated its effectiveness in the remote sensing field [21], [22]. On the other hand, the spatial information supplied by the VHSR image is integrated in the pipeline via the use of a Convolutional Neural Network, a more suitable family of models for spatial/contextual feature extraction [1].

The two branches of analysis learn complementary features that are successively combined for the land cover mapping, performed at the scale of the Sentinel-2 pixel. Following the idea proposed in [23] in which auxiliary classifiers were introduced with the aim to learn two sets of complementary features that are as much as possible discriminative when used alone; we also introduce two additional auxiliary classifiers, working independently on each branch of analysis, as shown in the Figure 1. A third classifier, working on the fusion (by concatenation) of the two sets of features, produce the final land use classification.

Each of the above mentioned classifiers is built by directly connecting the associated features to the output neurons on which the SoftMax activation function is successively applied [10]. The model weights are learned by back-propagation. The cost-function associated to the model is derived by a linear combination of the individual cost function of each of the classifiers.

### B. Integration of HSR time series information

Recurrent Neural Networks are well established machine learning techniques that demonstrate their quality in different domains such as speech recognition, signal processing, and natural language processing [24], [25]. Unlike standard feed forward networks (e.g., Convolutional Neural Networks – CNNs), RNNs explicitly manage temporal data dependencies since the output of the neuron at time t-1 is used, together with the next input, to feed the neuron itself at time t.

Recently, recurrent neural network (RNN) approaches have demonstrated their quality in the remote sensing field to produce land use mapping using time series of optical images [16] and recognize vegetation cover status using Sentinel-1 radar time series [22]. Motivated by these recent research results, we introduce an RNN module to integrate information from the Sentinel-2 time series into our fusion process. In our model, we choose the GRU unit (Gated Recurrent Unit) introduced by [20] since it has a moderate number of parameters to learn and it has already demonstrated its effectiveness in the field of remote sensing [16], [21]. We coupled the Gated Recurrent Unit with an *attention* mechanism [26].

The input of a RNN unit is a sequence of variables ($x_{t_1}$,..., $x_{t_N}$) where a generic element $x_{t_i}$ is a feature vector and $t_i$ refers to the corresponding time stamp. In the context of HSR satellite image time series, $x_{t_i}$ is a vector with as many components as the number of spectral bands (including raw

[1]Data are available via http://theia.cnes.fr, preprocessed in surface reflectance via the *MACCS-ATCOR Joint Algorithm* [19] developed by the National Centre for Space Studies (CNES).

[2]This was done to ensure a direct and non-overlapping correspondence between the time series pixels (10 m) and a block of VHSR pixels (5 × 5).

bands and indexes) carried by each satellite image. Equations 1, 2 and 3 formally describes the *GRU* neuron.

$$z_{t_i} = \sigma(W_{zx}x_{t_i} + W_{zh}h_{t_{i-1}} + b_z) \quad (1)$$

$$r_{t_i} = \sigma(W_{rx}x_{t_i} + W_{rh}h_{t_{i-1}} + b_r) \quad (2)$$

$$h_{t_i} = z_t \odot h_{t-1} + (1 - z_{t_i}) \odot \tanh(W_{hx}x_t + W_{hr}(r_t \odot h_{t_{i-1}}) + b_h) \quad (3)$$

The $\odot$ symbol indicates an element-wise multiplication while $\sigma$ and $\tanh$ represent Sigmoid and Hyperbolic Tangent function, respectively.

The *GRU* unit has two gates, update ($z_t$) and reset ($r_t$), and one cell state, i.e., the hidden state ($h_t$). Moreover, the two gates combine the current input ($x_t$) with the information coming from the previous timestamp ($h_{t-1}$). The update gate effectively controls the trade off between how much information from the previous hidden state will carry over to the current hidden state and how much information of the current timestamp needs to be kept. On the other hand, the reset gate monitors how much information of the previous timestamps needs to be integrated with current information. As all hidden units have separate reset and update gates, they are able to capture dependencies over different time scales. Units more prone to capturing short-term dependencies will tend to have a frequently activated reset gate, but those that capture longer-term dependencies will have update gates that remain mostly active [20]. This behavior enables the GRU unit to remember long-term information.

Attention mechanisms [26] are widely used in automatic signal processing (language or 1D signal) and they allow to gather together the information extracted by the GRU model at the different timestamps. The output returned by the GRU model is a sequence of feature vectors learned for each date: ($h_{t_1}$,..., $h_{t_N}$) where each $h_{t_i}$ has the same dimension $d$. Their matrix representation $H \in \mathbb{R}^{N,d}$ is obtained vertically stacking the set of vectors. The attention mechanism allows us to combine together these different vectors $h_{t_i}$, in a single one $rnn_{feat}$, to attentively combine the information returned by the GRU unit at each of the different timestamps. The attention formulation we use, starting from a sequence of vectors encoding the learned descriptors ($h_{t_1}$,..., $h_{t_N}$), is the following one:

$$v_a = tanh(H \cdot W_a + b_a) \quad (4)$$

$$\lambda = SoftMax(v_a \cdot u_a) \quad (5)$$

$$rnn_{feat} = \sum_{i=1}^{N} \lambda_i \cdot h_{t_i} \quad (6)$$

where matrix $W_a \in \mathbb{R}^{d,d}$ and vectors $b_a, u_a \in \mathbb{R}^d$ are parameters learned during the process. These parameters allow to combine the vectors contained in matrix $H$. The purpose of this procedure is to learn a set of weights ($\lambda_{t_1}$,..., $\lambda_{t_N}$) that allows the contribution of each time stamp to be weighted by $h_{t_i}$ through a linear combination. The $SoftMax(\cdot)$ [16] function is used to normalize weights $\lambda$ so that their sum is equal to 1. The output of the RNN module is the feature vector $rnn_{feat}$: they encode temporal information related to $ts_i$ for the pixel $i$.

### C. Integration of VHSR information

The VHSR information is integrated in $M^3Fusion$ through a CNN module. Computer vision literature offers several convolutional architectures [27], [28] aimed at classifying images. Most of these networks are designed to process RGB images (three channels) having size higher than 200x200 pixels. Such networks are composed by multiple (tens or hundreds) layers. In our scenario, the image patch has a size of 25x25 pixels and it involves five channels. In order to adopt a CNN module that well fits our scenario and remains computational affordable parameter-wise, we design the CNN module depicted in Figure 2.

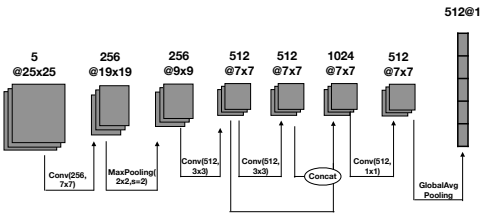



Figure 2: Convolutional Neural Network Structure

Our CNN network applies a first 7×7 kernel to the five-channel patch in order to produce 256 feature maps. Then, a *max pooling* layer is used to reduce the size and the number of parameters. Two successive convolution operations, with a 3×3 kernel, extract 512 feature maps each, which in their turn, are concatenated and reduced again by a convolution 1×1 kernel. The final size is then 512×7×7.

Finally, a Global Average Pooling operation enables the construction of a feature vector of size 512.

Each convolution is associated with a linear filter, followed by a Rectifier Linear Unit (ReLU) activation function [29] to introduce non-linearity and a batch normalization step [30]. The ReLU activation function is defined as follows:

$$ReLU(x) = Max(0, W \cdot x + b) \quad (7)$$

This activation function is defined on the positive part of the linear transformation of its argument ($W \cdot x + b$). The choice of ReLU nonlinearities is motivated by two factors: the good convergence properties it guarantees and ii) the low computational complexity it provides [29]. Furthermore, batch normalization [30] accelerates Deep Network training convergence by reducing the internal covariate shift.

The key points of our proposal are twofold: a) a higher number of filters in the first step and b) the concatenation of feature maps at different resolutions. The first point is related to the higher amount of spectral information (five channels) in input of our model compared to RGB images. To better exploit the high spectral richness of these data, we have increased the

number of feature maps generated at this stage. The second point concerns the concatenation of feature maps. With the goal of exploiting information at different resolutions we adopt a philosophy similar to [28] in which feature maps, at different level of the Deep architecture, are concatenated together. The output of this module is a vector of dimensions 512 ($cnn_{feat}$) which summarizes the spatial context ($patch_i$) associated to the $i$-th Sentinel-2 pixel.

### D. The End-To-End Fusion process

One of the advantages of deep learning, compared to standard machine learning methods, is the ability to link, in a single pipeline, the feature extraction step and the associated classifier [10]. This capability is particularly important in a multi-source, multi-scale and multi-temporal fusion process, such as the one represented by our scenario. $M^3Fusion$ leverages this characteristic to extract complementary knowledge from two data sources that describe the same information from different points of view. In addition, the combination/fusion of the data sources is optimized for the specific task at hand: land cover mapping. Our approach combines together (fuses) the heterogeneous spectral information belonging to the two data sources via multiple non-linear combination of the radiometric information.

To further strengthen the complementarity as well as the discriminative power of the learned features for each information branch, we adapt the technique proposed in [23] to our problem. In [23], the authors propose to learn two complementary representations (using two convolutional networks) from the same image. The discriminative power is enhanced by two auxiliary classifiers, linked to each group of features, in addition to the classifier that uses the merged information. The complementary is enforced by alternating the optimization of the parameters of the two branches. In our case, we have two complementary sources of information (sentinel-2 time series and VHSR data) to which two auxiliary classifiers are connected to independently increase their ability to discriminate among land cover classes.

In detail, the classifier that exploits the full set of features is fed by concatenating the output features of both CNN ($cnn_{feat}$) and RNN ($rnn_{feat}$) modules together. Empirically, we have observed that the RNN module overfits the data. To alleviate this problem, we add a Dropout layer [31] on $rnn_{feat}$ with a drop-rate equals to 0.4. The learning process involves the optimization of three classifiers at the same time, one specific to $rnn_{feat}$, a second one related to $cnn_{feat}$ and the third one that considers $[rnn_{feat}, cnn_{feat}]$.

The cost function associated to our model is :

$$\begin{aligned} L_{total} &= \alpha_1 * L_1(rnn_{feat}, W_1, b_1) + \\ &= \alpha_2 * L_2(cnn_{feat}, W_2, b_2) + \\ &= L_{fus}([cnn_{feat}, rnn_{feat}], W_3, b_3) \end{aligned} \quad (8)$$

where

$$L_i(feat, W_i, b_i) = L_i(Y, SoftMax(feat \cdot W_i + b_i))$$

with $Y$ being the true value of the class variable. $L_1(rnn_{feat}, W_1, b_1)$ (resp. $L_2(cnn_{feat}, W_2, b_2)$) is the cost function of the first (resp. the second) auxiliary classifier that takes as input the set of descriptors returned by the RNN module (resp. CNN module) and the parameters $W_1, b_1$ (resp. $W_2, b_2$) to make the prediction. $L_{fus}(cnn_{feat}, rnn_{feat}, W_3, b_3)$ is the cost function of the classifier that uses the combined set of features ($[cnn_{feat}, rnn_{feat}]$). This last cost function is parameterized through $W_3$ et $b_3$. Each cost function is modeled through categorical cross entropy, a typical choice for multi-class supervised classification tasks [16].

$L_{total}$ is optimized end-to-end. Once the network has been trained, the prediction is carried out only by means of the classifier involving $W_3$ and $b_3$, which uses all the features learned by the two branches. The cost functions $L_1$ et $L_2$, as highlighted in [23], operate a kind of regularization that forces, within the network, the features extracted to be discriminative independently.

We underline that the data fusion step is achieved internally by the proposed deep learning architecture without the necessity to resample images at the same spatial and spectral resolution.

## IV. Experiments

In this section, we present and discuss the experimental results obtained on the study site introduced in Section II.

In the evaluation, we investigate several points deeply related to a more clear understanding of our framework. As first point, we perform an in-depth evaluation of the performance of our proposal. In this part we assess: i) how the performances of the CNN module change varying the amount of spatial information considered; ii) the benefit of forcing source specific features to be as much as possible discriminative by themselves, and iii) the effectiveness of the fusion method with respect to the use of each information source alone. The second set of experiments are more related to the comparison between the ability of $M^3Fusion$ and a standard machine learning classifier (***Random Forest***)to deal with land cover mapping. In this part we evaluate: i) the per-class performance of our framework compared to the one of the competitor and ii) the robustness of $M^3Fusion$ considering different splits of the original datasets. Finally, in the last part of the evaluation, we perform a qualitative study considering the maps obtained by the competing methods. This evaluation supplies some examples that support the quality and effectiveness of our framework.

### A. Experimental Settings

Here we describe the implementation details of $M^3Fusion$ and the competitor we will use in Section IV-E, namely Random Forest classifier (***RF***), which is commonly used for supervised classification in the field of remote sensing [9].

For the ***RF*** model, we fix the number of generated random trees to 200. We use the publicly available Python implementation supplied by the *scikit-learn* library [32]. In order to fairly compare the two methods, we supply the same input data set both to ***RF*** and to $M^3Fusion$. Each example of the data set

provided to the Random Forest approach has a size of 3 669, corresponding to $25 \times 25 \times 5$ ($patch_i$) plus $34 \times 16$ ($ts_i$).

In our model, we choose the value $d$ (number of hidden units for the RNN module) equals to 1 024. We empirically fix $\alpha_1$ and $\alpha_2$ to 0.3. During the learning phase, we use the Adam method [33] to learn the model parameters with a learning rate equal to $2 \cdot 10^{-4}$. The training process is conducted over 400 epochs. The model that reaches the lowest value of the cost function (at training time) is used in the test phase.

$M^3Fusion$ is implemented using the Python Tensorflow library. The learning phase takes about 15 hours while the classification on the test data takes about one minute on a workstation with an Intel (R) Xeon (R) CPU CPU E5-2667 CPU v4@3.20Ghz with 256 GB of RAM and TITAN X GPU.

The data are prepared as follows. We divide the dataset into two parts, one for learning and the other one to test the performance of the supervised classification methods. We used 30% of the objects for the training phase (97 110 pixels - 797 objects) while the remaining 70% are used for the test phase (225 638 pixels - 1 859 objects). We impose that pixels of the same object belong exclusively to the training or to the test set [8]. The values are normalized, per spectral band, in the interval $[0, 1]$.

Finally, the assessment of the classification performances is done considering global precision (*Accuracy*), *F-Measure* [16] and Kappa.

### B. Evaluating the Patch Size for the CNN Module

The first experiment we conduct is meant to understand the influence of the patch size considering the CNN module. To this purpose, we vary the patch size considering the following different values: 15x15, 25x25, 35x35 and 45x45.

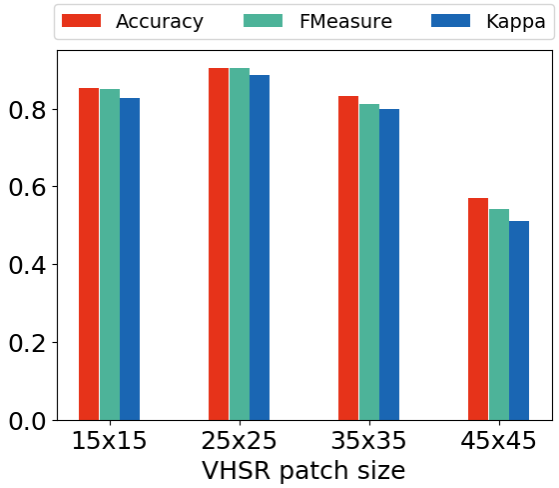



Figure 3: Accuracy, F-Measure and Kappa of the four different VHSR patch size

Figure3 shows the Accuracy, F-Measure and Kappa resulting from the application of the four different patch sizes to our CNN module, considering only the VHSR information. We observe that high values of patch size (i.e., 45x45), instead of improving the CNN performance, degrade the final results. Applying patches of size 45x45 is equivalent to analyze areas of 90m x 90m. Probably, what is happening is that the spatial context is too wide and the information contained in this patch is not discriminant enough to characterize the spatial context of a particular kind of land cover. Reducing the spatial patch to areas of 30m x 30m (15x15) or 50m x 50m (25x25) will supply more discriminative information for the characterization of the land cover. This experiment suggests that the patch needs to be accurately chosen considering the particular task at hand.

### C. Assessment of the Impact of Auxiliary Classifiers

Another component that characterizes the $M^3Fusion$ model is the use of auxiliary classifiers in order to strengthen the discriminative power of each set of learned features independently [23]. With the aim of validating the importance of the auxiliary classifier within our model, we perform an experiment consisting in the comparison of $M^3Fusion$ with a modified version deprived of the auxiliary classifiers, named $M^3Fusion - NoAux$. Figure 4 reports the F-Measure per class of this comparative analysis.

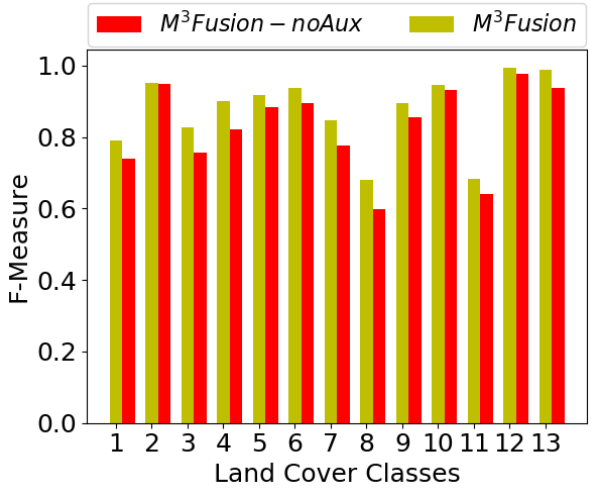



Figure 4: Per-Class F-Measure for $M^3Fusion$ with and without auxiliary classifiers.

Generally, we note that the version with auxiliary classifiers outperforms $M^3Fusion-NoAux$ no matter which land cover class is considered. This result underlines the importance of boosting the independent discriminative power of the learned features (per source) as much as possible before fusing them to perform the final classification.

### D. $M^3Fusion$ vs. CNN vs. RNN

$M^3Fusion$ leverages both spatial and temporal information with the aim of improving the land cover mapping task. Here, we investigate whether the use of both VHSR and SITS information, together, can effectively improve the final land cover mapping. To assess this point, we evaluate the performances of $M^3Fusion$ compared to the ones achieved by the two modules (CNN and RNN) independently. To this purpose, we proceed as follows. For each type of information (VHSR and SITS, respectively), we train a CNN (resp. RNN) with the

same structure as the corresponding branch in $M^3Fusion$. As done before, we report the results in terms of F-Measure per-class to understand how the different classifiers behave on recognizing the different land cover classes involved in our task. Figure 5 visually summarizes the F-Measure results.

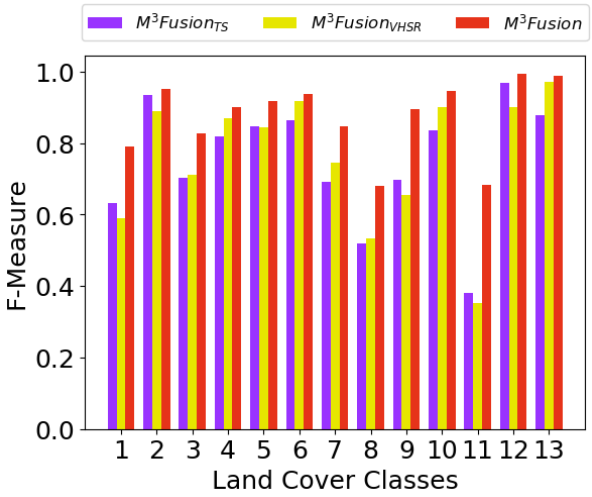



Figure 5: F-Measure by class of both $M^3Fusion_{ts}$ and $M^3Fusion_{vhsr}$ branches, compared to $M^3Fusion$ method.

As we can note, the $M^3Fusion$ approach clearly outperforms the individual classifiers that use only one type of information. We observe that, systematically, the fusion approach effectively exploits the complementarity of the two sources of information improving the final classification performances. This phenomenon is particularly clear for some classes such as (1),(7),(8), (9) and (11) (resp. *Crop Cultivations*, *Shrubby savannah*, *Herbaceous savannah*, *Bare rocks* and *Greenhouse crops*) where the gain in F-Measure is higher than 0.15 points. A possible explanation is that the detection of the different type of savannah as well as the crop culture cannot be efficiently reached without considering, simultaneously, the temporal evolution of the spectral profile as well as the spatial context.

### *E. Comparative Analysis*

Figure 6 provides the results of a comparative analysis between our model and Random Forest (RF), an ensemble learning method that is commonly employed in the field of Remote Sensing for dealing with the land cover mapping task.

We observe that $M^3Fusion$ reaches higher performance indices than Random Forest on all the land cover classes. The highest gains are related to classes (1), (3), (9), (10) and (11) (resp. *Crop Cultivation*, *Orchards*, *Bare rocks*, *Urbanized areas* and *Greenhouse crops*). Considering the characteristics of such classes, the different gains are the results of the effectiveness of $M^3Fusion$ to combine temporal and fine spatial information together leveraging the complementary of the two sources of information. With the aim of better understanding the misclassification behavior of the two approaches, we report in Figure 7 the confusion matrix of both $M^3Fusion$ and RF. A closer look at these statistics points out that $M^3Fusion$ is

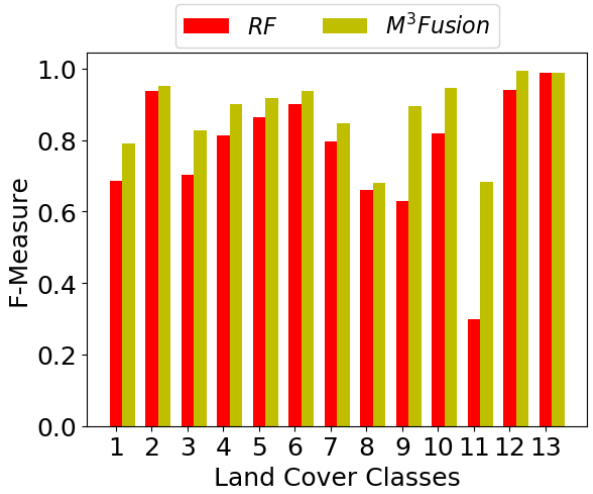
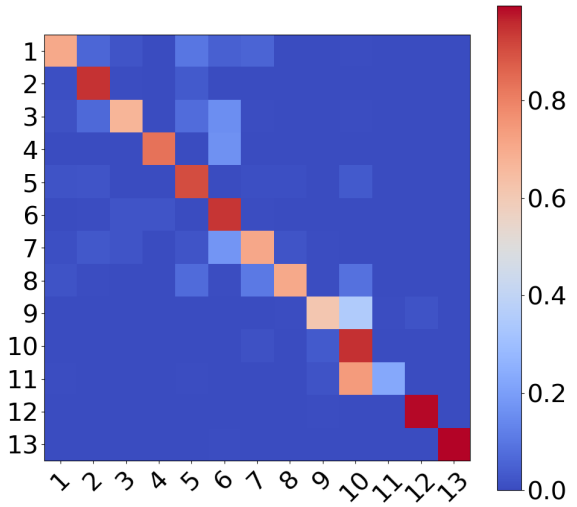



Figure 6: Per class F-Measure results of Random Forest and $M^3Fusion$ methods.

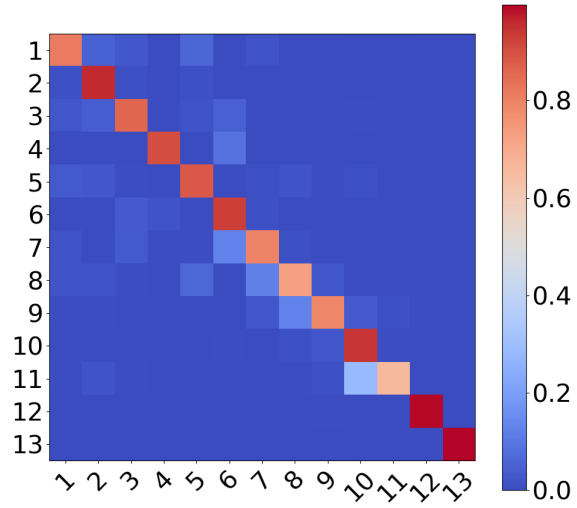
more precise than the competitor. This consideration emerges from the fact that the corresponding heat map (Figure 7b) has a more visible diagonal structure (the dark red blocks concentrated on the diagonal). This is not the case for Random Forest (Figure 7a) where the distinction between classes is less sharp.

This behavior is particularly visible for the *Greenhouse crops* land cover (class (11) ), where the majority of the elements belonging to this class are categorized as *Urbanized areas* (class (10)). A similar phenomena affecting the performance of the *Random Forest* classifier can be observed between *Bare rocks* and *Urbanized area* classes. On the other hand, $M^3Fusion$ tends to have some confusion on these classes too, but the extent of this phenomenon is attenuated with respect to the *Random Forest* method. Notice that these classes represent land cover that have very similar temporal radiometric behavior but they can be characterized by different spatial context; this spatial context is intelligently leveraged by the fusion process performed by $M^3Fusion$ to reduce the misclassification error.

As further comparative analysis, in Table II we report a summary of the results obtained by applying the two approaches ($M^3Fusion$ and RF) on the fusion of the two information sources (VHSR and SITS) as well as on each single source pf information individually. In the latter case, the approaches are named as follows: $RF_{ts}$ and $M^3Fusion_{ts}$ stand for RF and $M^3Fusion$ applied on time series data only; $RF_{vhsr}$ and $M^3Fusion_{vhsr}$ stand for RF and $M^3Fusion$ applied on VHSR data only. This analysis is similar to the one presented in Section IV-D, but here we also evaluate the Random Forest classifier on each source separately. We compare the different methods by means of Accuracy, (average) F-Measure and Kappa. As we previously observed in Section IV-D, also for the Random Forest approach the use of multiple sources results in a general improvement of land cover mapping performances. This behavior points out once again that the two sources of information carrying out complementary knowledge




(a)




(b)

Figure 7: Heat Maps representing the confusion matrices of (a) Random Forest and (b) $M^3Fusion$.

and the joint use of temporal and fine spatial information positively influence the land cover classification task.

After a closer look at the values of all the evaluation metrics, we can state that, on the *Reunion Island* dataset, the data fusion process implemented by $M^3Fusion$ is more effective than the one carried out by the Random Forest classifier. The Deep Learning data fusion approach smartly leverages the complementary information reaching a gain of more than 0.06 Accuracy point with respect to the best individual source application scenario ($M^3Fusion_{vhsr}$) while, in the case of Random Forest, this gain is limited to less than 0.02 Accuracy points compared to its best individual source result ($RF_{ts}$).

### *F. Robustness of $M^3Fusion$*

The results presented so far are related to a single 30%/70%-split of our data set. It is known that, depending on the split of the data, the performances of the different methods may vary as simpler or more difficult examples are involved in the training or test set. With the objective of understanding the robustness of our method with respect to this phenomenon, we build four different random 30%/70%-splits of the dataset, using the same protocol described in Section IV-A. The results achieved by $M^3Fusion$ and RF on the five splits are shown in the Table III.

| | *Accuracy* | *F-Measure* | *Kappa* |
|---|---|---|---|
| $RF_{ts}$ | 0.8543 | 0.8519 | 0.8258 |
| $M^3Fusion_{ts}$ | 0.8319 | 0.8325 | 0.8033 |
| $RF_{vhsr}$ | 0.8237 | 0.8140 | 0.7908 |
| $M^3Fusion_{vhsr}$ | 0.8369 | 0.8364 | 0.8677 |
| $RF$ | 0.8716 | 0.8681 | 0.8491 |
| $M^3Fusion$ | **0.9149** | **0.9148** | **0.9000** |

Table II: Accuracy, F-Measure, Kappa of different methods considering the fusion process as well as one source at time

| | *RF* | | | $M^3Fusion$ | | |
|---|---|---|---|---|---|---|
| | *Accuracy* | *F-Measure* | *Kappa* | *Accuracy* | *F-Measure* | *Kappa* |
| 0 | 0.8772 | 0.8737 | 0.8536 | 0.9114 | 0.9112 | 0.8963 |
| 1 | 0.8759 | 0.8720 | 0.8521 | 0.8950 | 0.8954 | 0.8771 |
| 2 | 0.8716 | 0.8681 | 0.8491 | 0.9149 | 0.9148 | 0.9000 |
| 3 | 0.8824 | 0.8790 | 0.8625 | 0.9114 | 0.9107 | 0.8953 |
| 4 | 0.8757 | 0.8708 | 0.8536 | 0.9061 | 0.9051 | 0.8899 |

Table III: Accuracy, F-Measure, Kappa on different random splits

We note that $M^3Fusion$, always supplies better performances, in terms of all the involved metrics, than Random Forest does. We observe that the performances of $M^3Fusion$, in terms of accuracy, vary between 0.8950 and 0.9149 while, those achieved by Random Forest vary between 0.8716 and 0.8824. It is worth noting that the best accuracy value observed for Random Forest (0.8824) is lower than the worst accuracy achieved by our approach (0.8950).

| | *Gain* | | |
|---|---|---|---|
| | *Accuracy* | *F-Measure* | *Kappa* |
| 0 | +0,0342 | +0,0374 | +0,0426 |
| 1 | +0,0191 | +0,0234 | +0,0250 |
| 2 | **+0,0433** | **+0,0466** | **+0,0508** |
| 3 | +0,0289 | +0,0317 | +0,0328 |
| 4 | +0,0303 | +0,0343 | +0,0362 |

Table IV: $M^3Fusion$'s gain in term of Accuracy, F-Measure, Kappa for each split

Finally, in Table IV, we report the gain of $M^3Fusion$ with respect to Random Forest considering the whole set of evaluation metrics. It is worthy of note that the gain in F-Measure and Kappa is always higher than the gain in Accuracy. This phenomenon indicates that not only $M^3Fusion$ outperforms the competitor on the majority (well represented) classes but, it exhibits better performances on all the land cover classes, independently if they are well represented or not. From a closer look at the results, $RF$ emerges as being more influenced by class imbalance, giving more chance to the highly represented classes in its decisions. For instance, this phenomena happens between *Greenhouse crops* (low represented) and *Urbanized areas* (high represented) as well as *Orchards* (low represented) and *Forest* (high represented) land cover classes (Figure 7).

### G. Map Comparison

In addition to the evaluations reported in the previous sections, we also propose a first visual qualitative evaluation of the produced maps. The maps obtained by $M^3Fusion$ (resp. Random Forest) is shown in Figure 8b (resp. Figure 8a) for a qualitative overview. When we visually analyze the map issued by $M^3Fusion$, we observe that the detection of the majority classes, i.e., the areas cultivated with sugar cane on the coast, as well as the various natural areas (grasslands, savannas and forests) and the urban areas are well recognized with less salt and pepper error than the map produced by the *Random Forest* classifier (Figure 8a).

Some comparisons between the two maps are provided at the scale of some remarkable details in Figure 9: in the first column, a fragment of urban areas is displayed, where the presence of noise is particularly marked for the $RF$'s map (in the middle). This phenomenon is highlighted by the transition zones between buildings, which are often interpreted as crops. This effect is less present on the $M^3Fusion$'s map (bottom). A particular interesting effect concerns the artifacts of the $RF$ map due to the presence of clouds or shadows (detail on second column) on the VHSR image, which are definitely mitigated in the map produced by $M^3Fusion$ for this example. This effect is also visible on larger cloudy areas (last column), where some errors persist in $M^3Fusion$'s map but most of the affected area is correctly retrieved. A possible explanation for these aberrations could be a biased prediction behavior of $RF$ in favor of information coming from VHSR data. Notice that this situation does not occur when the same data are processed by $M^3Fusion$. This behavior can be explained by the way the *Random Forest* works and the feature cardinality of each data source. Due to the random nature of *Random Forest*, each time it samples a random set of features to build the trees belonging to the forest. Since the number of features available from the VHSR source (3 125) is bigger than the number of features coming from the time series data (544); *Random Forest* tends to exploit more frequently features coming from the former source than from the latter one. This fact probably bias the *Random Forest* to overuse VHSR information.

A last example showing map improvements is on the third column of Fig. 9, where the dense urban area has reduced noise in $M^3Fusion$'s map with respect to $RF$'s one, and some clear errors are corrected on vegetated areas (e.g. grass among airport lanes is erroneously classified as sugar cane using $RF$, while $M^3Fusion$mostly detects the meadows class).

### H. Discussion

The experimental results, both quantitative and qualitative, have demonstrated the ability of the proposed deep learning architecture to cope with the issue of land cover mapping from multiple remote sensing data sources acquired at different spatial/spectral/temporal resolution. $M^3Fusion$ has shown improvements all over the considered land cover classes involved in the *Reunion* dataset compared to the performances of the *Random Forest* classifier. The biggest gain in performance is related to the *Greenhouse crop* class. The competing approach has serious issue to recognize this class and distinguish it from the *Urbanized Areas*. On the other hand, $M^3Fusion$ demonstrates the ability to improve the results on the *Greenhouse crop* class. Similar behavior is exhibited considering the *Bare rocks* and *Urbanized Areas* land cover classes. $M^3Fusion$ exploits the spatial context information (supplied by the SPOT6 image) to differentiate between the *Greenhouse crop* and *Urbanized Areas* classes. Probably, the area surrounding a *Greenhouse crop* pixel is quiet different from the area surrounding a general urban area. The Deep Learning architecture we proposed, via the CNN module, is able to leverage this (contextual) information while *Random Forest* does not.

In addition, the *Random Forest* method seems to be biased towards high represented classes (i.e. *Forest*, *Urbanized Areas* due to the unbalanced nature of the dataset and towards VHSR missing information (i.e. cloudy phenomena in the SPOT6 image). This last point is probably related to the fact that VHSR information constitutes around 85% of the input information (3 125 over 3 669 features). Conversely, $M^3Fusion$ exhibits a more stable behavior considering the issue related to the high/low represented land cover classes improving performances on such categories. In addition, preliminary in-depth analysis of produced maps underlining that $M^3Fusion$ is capable to alleviate issue related to one of the data sources (i.e. missing data) leveraging information from the other one. This behavior highlights the ability of the deep learning architecture to exploit the complementarity between data sources to deal with the task of land cover mapping from VHSR SPOT6 image and a time series of Sentinel-2 satellite images.

Considering a more close analysis of $M^3Fusion$, we have noted that the dimension of the VHSR patches used to feed the convolution branch needs to be carefully chosen. Experiments have shown that big patches negatively influence the behavior of the neural approach since, probably, they introduce too much contextual confusion in the learning process. Our suggestion is to use some knowledge about the study area to reasonably choose the patch size. However, possible extensions of $M^3Fusion$ can be related on how integrate data sources at their native resolution. Currently, both SPOT6 and Sentinel-2 information are exploited after a resample step. In the case of SPOT6, PAN and MS information are combined via pansharpening while 20 meters Sentinel-2 bands are resampled at 10 to have coherent per source spatial information. A step further towards complete, land cover oriented, data fusion of SPOT6 and Sentinel 2 images will be the direct integration of radiometric information at their native resolution.

## V. Conclusions

In this article, we have proposed a new deep learning architecture for the fusion of satellite data at high temporal/spatial resolution with an image at very high spatial resolution (VHSR) to perform land cover mapping. Experiments carried out on a study site have validated the quality and effectiveness of our approach compared to a common machine learning approach usually employed in the field of remote sensing. In

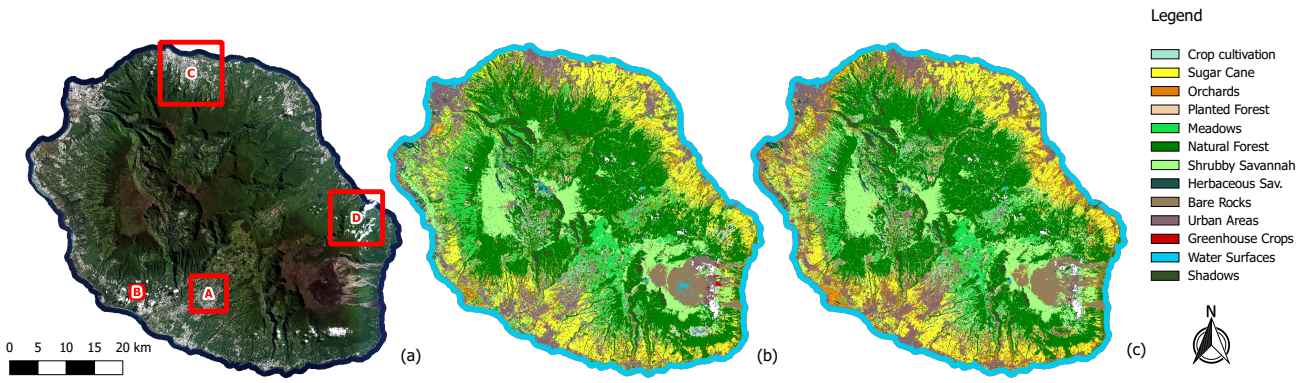



Figure 8: Source VHRS scene (a) (see Fig. 9 for details in red boxes), maps produced by $RF$ (a) and $M^3Fusion$ (b).

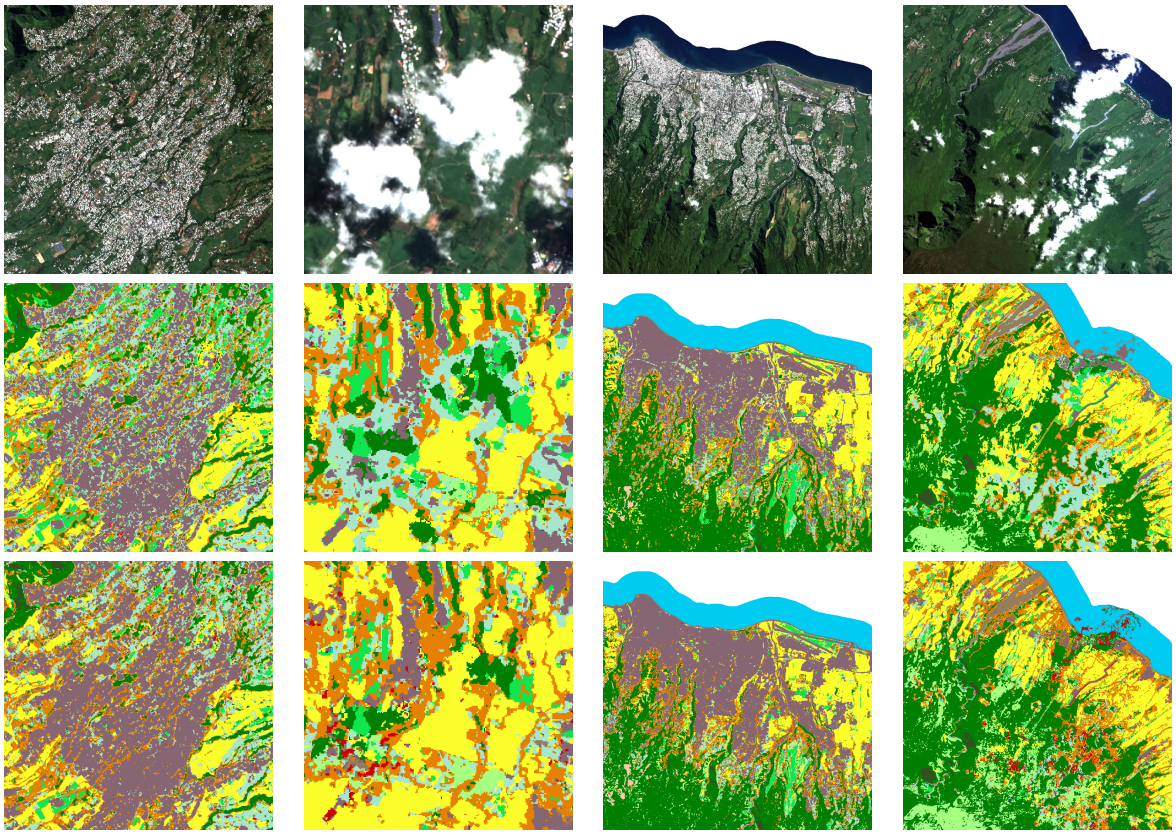


Figure 9: Classification results obtained with $RF$ and $M^3Fusion$. Top to bottom: excerpts from SPOT6/7 imagery (respectively A,B,C,D from Fig. 8), classification by $RF$, classification by $M^3Fusion$.

the future, we plan to investigate several extensions of our architecture to integrate other complementary data sources. Another possible future development will be the exploitation of the data sources at their original resolution. Currently, both VHSR and Time Series information are resampled at different spatial resolutions introducing some possible bias. Use data sources at their native resolution can avoid useless preprocessing and, probably, increase classification performances.

This work has been supported by

## VI. Acknowledgements

This work was supported by the French National Research Agency under the Investments for the Future Program, referred as ANR-16-CONV-0004 (DigitAg), the GEOSUD project with reference ANR-10-EQPX-20 and the Programme National de Télédétection Spatiale (PNTS, http://www.insu.cnrs.fr/pnts ), grant $n^o$PNTS-2018-5, as well as from the financial contribution from the French Ministry of agriculture "Agricultural and Rural Development" trust account. This work also used an image acquired under the CNES Kalideos scheme (La Réunion site).

## REFERENCES


[1] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Convolutional neural networks for large-scale remote-sensing image classification," *IEEE TGRS*, vol. 55, no. 2, pp. 645–657, 2017.

[2] N. A. Abade, O. A. d. C. Jnior, R. F. Guimares, and S. N. de Oliveira, "Comparative analysis of modis time-series classification using support vector machines and methods based upon distance and similarity measures in the brazilian cerrado-caatinga boundary," *Remote Sensing*, vol. 7, no. 9, pp. 12 160–12 191, 2015.

[3] M. Schmitt and X. X. Zhu, "Data fusion and remote sensing: An ever-growing relationship," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 4, pp. 6–23, 2016.

[4] A. Karpatne, Z. Jiang, R. R. Vatsavai, S. Shekhar, and V. Kumar, "Monitoring land-cover changes: A machine-learning perspective," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, pp. 8–21, 2016.

[5] F. Gao, J. G. Masek, M. R. Schwaller, and F. G. Hall, "On the blending of the landsat and MODIS surface reflectance: predicting daily landsat surface reflectance," *IEEE Trans. Geoscience and Remote Sensing*, vol. 44, no. 8, pp. 2207–2218, 2006.

[6] K. Knauer, U. Gessner, R. Fensholt, and C. Kuenzer, "An ESTARFM fusion framework for the generation of large-scale time series in cloud-prone and heterogeneous landscapes," *Remote Sensing*, vol. 8, no. 5, p. 425, 2016.

[7] Y. T. S. Correa, F. Bovolo, and L. Bruzzone, "VHR time-series generation by prediction and fusion of multi-sensor images," in *2015 IEEE IGARSS*, 2015, pp. 3298–3301.

[8] J. Inglada, A. Vincent, M. Arias, B. Tardy, D. Morin, and I. Rodes, "Operational high resolution land cover map production at the country scale using satellite image time series," *Remote Sensing*, vol. 9, no. 1, p. 95, 2017.

[9] V. Lebourgeois, S. Dupuy, E. Vintrou, M. Ameline, S. Butler, and A. Bégué, "A combined random forest and OBIA classification scheme for mapping smallholder agriculture at different nomenclature levels using multisource data (simulated sentinel-2 time series, VHRS and DEM)," *Remote Sensing*, vol. 9, no. 3, p. 259, 2017.

[10] L. Zhang and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, pp. 22–40, 2016.

[11] G. Scarpa, M. Gargiulo, A. Mazza, and R. Gaetano, "A cnn-based fusion method for feature extraction from sentinel data," *Remote Sensing*, vol. 10, no. 2, p. 236, 2018.

[12] X. Xu, W. Li, Q. Ran, Q. Du, L. Gao, and B. Zhang, "Multisource remote sensing data classification based on convolutional neural network," *IEEE TGRS*, vol. 56, no. 2, pp. 937–949, 2018.

[13] X. Liu, L. Jiao, J. Zhao, J. Zhao, D. Zhang, F. Liu, S. Yang, and X. Tang, "Deep multiple instance learning-based spatial-spectral classification for PAN and MS imagery," *IEEE Trans. Geoscience and Remote Sensing*, vol. 56, no. 1, pp. 461–473, 2018.

[14] X. Zhu, D. Tuia, L. Mou, G. X. L. Zhang, F. Xu, and F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, p. 836, 2017.

[15] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE TPAMI*, vol. 35, no. 8, pp. 1798–1828, 2013.

[16] D. Ienco, R. Gaetano, C. Dupaquier, and P. Maurel, "Land cover classification via multitemporal spatial data by deep recurrent neural networks," *IEEE GRSL*, vol. 14, no. 10, pp. 1685–1689, 2017.

[17] D. H. T. Minh, D. Ienco, R. Gaetano, N. Lalande, E. Ndikumana, F. Osman, and P. Maurel, "Deep recurrent neural networks for winter vegetation quality mapping via multitemporal SAR sentinel-1," *IEEE Geosci. Remote Sensing Lett.*, vol. 15, no. 3, pp. 464–468, 2018.

[18] J. Geng, H. Wang, J. Fan, and X. Ma, "SAR image classification via deep recurrent encoding neural networks," *IEEE Trans. Geoscience and Remote Sensing*, vol. 56, no. 4, pp. 2255–2269, 2018.

[19] O. Hagolle, M. Huc, D. Villa Pascual, and G. Dedieu, "A Multi-Temporal and Multi-Spectral Method to Estimate Aerosol Optical Thickness over Land, for the Atmospheric Correction of FormoSat-2, LandSat, VEN$\mu$S and Sentinel-2 Images," *Remote Sensing*, vol. 7, no. 3, pp. 2668–2691, 2015.

[20] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP*, 2014, pp. 1724–1734.

[21] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE TGRS*, vol. 55, no. 7, pp. 3639–3655, 2017.

[22] D. H. T. Minh, D. Ienco, R. Gaetano, N. Lalande, E. Ndikumana, F. Osman, and P. Maurel, "Deep recurrent neural networks for winter vegetation quality mapping via multitemporal sar sentinel-1," *IEEE GRSL*, vol. Preprint, no. -, pp. –, 2018.

[23] S. Hou, X. Liu, and Z. Wang, "Dualnet: Learn complementary features for image recognition," in *IEEE ICCV*, 2017, pp. 502–510.

[24] K. Soma, R. Mori, R. Sato, N. Furumai, and S. Nara, "Simultaneous multichannel signal transfers via chaos in a recurrent neural network," *Neural Computation*, vol. 27, no. 5, pp. 1083–1101, 2015.

[25] T. Linzen, E. Dupoux, and Y. Goldberg, "Assessing the ability of lstms to learn syntax-sensitive dependencies," *TACL*, vol. 4, pp. 521–535, 2016.

[26] D. Britz, M. Y. Guan, and M. Luong, "Efficient attention using a fixed-size memory representation," in *EMNLP*, 2017, pp. 392–400.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[28] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017, pp. 2261–2269.

[29] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML10*, 2010, pp. 807–814.

[30] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[31] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *ICASSP*, 2013, pp. 8609–8613.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.