$M^3Fusion$: A Deep Learning Architecture for Multiscale Multimodal Multitemporal Satellite Data Fusion

Paola Benedetti, Dino Ienco, Raffaele Gaetano, Kenji Ose, Ruggero G. Pensa, Stéphane Dupuy

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

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

Abstract

Modern Earth Observation systems provide sensing data at different temporal and spatial resolutions. Among optical sensors, today the Sentinel-2 program supplies high-resolution 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 images (VHSR) are still an essential tool to figure out land cover mapping characterized by fine spatial patterns. Understand how to efficiently leverage these complementary sources of information together to deal with land cover mapping is still challenging.

With the aim to tackle land cover mapping through the fusion of multi-temporal High Spatial Resolution and Very High Spatial Resolution satellite images, we propose an End-to-End Deep Learning framework, named $M^3Fusion$, able to leverage simultaneously the temporal knowledge contained in time series data as well as the fine spatial information available in VHSR information. Experiments carried out on the *Reunion Island* study area asses 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 systems produce huge volumes of data every day. This information can be organized into time series of high-resolution satellite imagery (SITS) (i. e. Sentinel) that are useful for area monitoring over time.

In addition to this high temporal frequency information, we can also obtain Very High Spatial Resolution (VHSR) information, such as Spot6/7 or Pleiades imaging, with a more limited temporal frequency [1] (e. g. once a year).

The analysis of time series and its coupling/fusion with punctual VHSR data remains an important challenge in the field of remote sensing. [2], [3].

In the context of land use classification, employing high spatial resolution (HSR) time series, instead of a single image of the same resolution, can be useful to distinguish classes according to their temporal profiles [4]. On the other hand, the use of fine spatial information helps to differentiate other kind of classes that need spatial context information at higher scale [3].

Typically, the approaches that use these two types of information [5], [6], perform data fusion at descriptor level [3]. This type of fusion involves extracting a set of independent features for each data source (time series, VHSR image) and then stacking these features together to feed a traditional supervised learning method (i. e., Random Forest).

Recently, the deep learning revolution [7] has shown that neural network models are well adapted tools for automatically managing and classifying remote sensing data [7]. The main characteristic of this type of model is the ability to simultaneously extract features optimized to 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 and/or Pleiades).

Considering deep learning methods, we can find two main families of approaches: convolutional neural networks [7] (CNN) and recurrent neural networks [8] (RNN). CNN are well suited to model the spatial autocorrelation available in an image, while RNN networks are especially tailored to manage time dependencies [9] from multidimensional time series.

In this article, we propose to leverage both CNN and RNN to address the fusion problem between an HSR time series of Sentinel-2 images and a VHSR image on the same study area with the goal to perform land use mapping. The method we propose, named $M^3Fusion$ (Multi-Scale/Modal/Temporal Fusion), consists in a deep learning architecture that integrates both a CNN component (to manage VHSR information) and an RNN component (to analyze HSR time series information) in an end-to-end learning process. Each information source is integrated through its dedicated module and the extracted descriptors are then concatenated to perform the final classification. Setting up such a process, which takes both data sources into account at the same time, ensures that we can extract complementary and useful features for land use mapping.

To validate our approach, we conducted experiments on a data set involving the Reunion Island study 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 and conclusions are drawn in Section V.

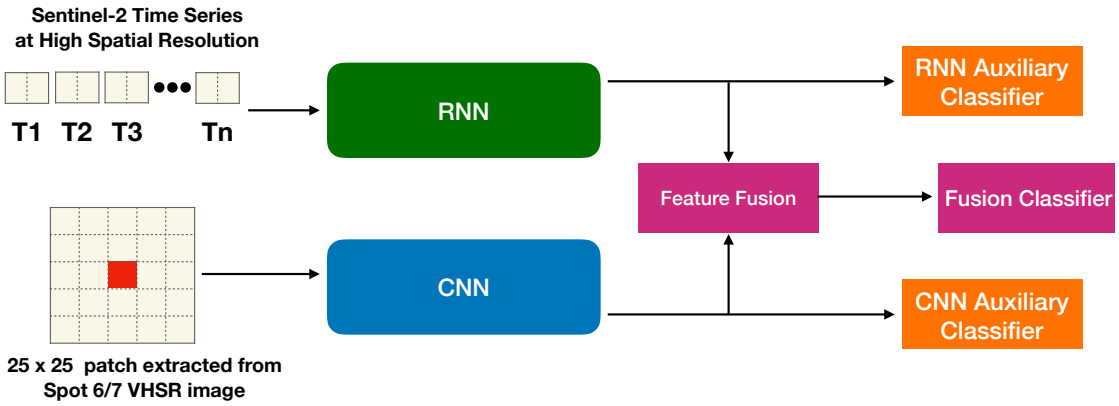


Figure 1: Visual representation of $M^3Fusion$.

II. Data

The study was carried out on 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 image (VHSR) SPOT6/7 acquired in April 2016 and covering the whole island. The S2 images used are those provided at level 2A by the Continental Surfaces pole THEIA[1], where the bands at 20 m resolution were resampled to 10 m. A preprocessing was performed to fill cloudy observations through a linear multi-temporal interpolation over each band (cfr. *Temporal Gapfilling*, [5]), 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) [5], [6]). 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, originally consisting of a 1.5 m panchromatic band and 4 multispectral bands (blue, green, red and near infrared) at 6 m resolution, was merged to produce a single multispectral image at 1.5 m resolution and then resampled at 2 m because of the network architecture learning requirements.[2]. Its final size is 33280 $\times$ 29565 pixels on 5 bands (4 reflectors *Top of Atmosphere* plus the NDVI). This image was also used as a reference to realign the different images in the time series by a 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 graphical parcel register (RPG) data set 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 natural and urban spaces . All polygon contours have been resumed using the VHSR image as a reference. The final dataset includes a total of 322 748 pixels (2 656 objects) distributed over 13 classes, as indicated in the Table I.

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

Table I: Characteristics of the Reunion Dataset

III. CONTRIBUTIONS

A. $M^3Fusion$ model description

Figure 3 visually describes the $M^3Fusion$ approach proposed in this article. First of all, we define the input data for our deep learning model. $M^3Fusion$ takes as input a data $\{(x_i, y_i)\}_{i=1}^{M}$ 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 (10m resolution) and $patch_i$ is a subset of the image Spot6/7 (2m resolution) centered around the corresponding pixel Sentinel-2. Every example has two representation: a temporal HSR modality (provided by the Sentinel-2 time series) and a VHSR modality (provided by the Spot6/7 image).

For every $patch_i$, we fixed the window size to 25×25 pixels on the Spot6/7 (which correspond 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 with the VHSR one contained in the Spot 6/7 image, we designed a network which has two parallel branches, one for each of the two modes (spatial/temporal). For the time series concerning the Sentinel-2 pixel we use a Recurrent Neural Network (RNN) architecture. In particular, we used a Gated Recurrent Unit (GRU) introduced in [11] which has already demonstrated its effectiveness in the remote sensing field [12], [13]. On the other hand, the spatial information, with a scale of 2m, introduced through VHSR image is integrated through the use of a Convolutional Neural Network [1] which allows to extract spatial context knowledge around the Sentinel-2 pixel.

Via the two streams of analysis we learn two complementary groups of features that we successively leverage for the classification that is performed at the scale of the Sentinel-2 pixel. According to the philosophy introduced in [14], the proposed architecture aims to learn two sets of complementary features (thanks to the different spatial and temporal modalities) that are as much as possible discriminative when used alone. To ensure this last point, the strategy provides in [14] introduces two additional auxiliary classifiers, working independently on each group of features, as shown in the Figure 3. A third classifier, working on the fusion (by concatenation) of the two sets of features, perform the final land use mapping.

Each of the above mentioned classifiers is realized by directly connecting the associated features to the output neurons on which SoftMax activation function is successively applied [7]. 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 information from the HSR time series

Recently, recurrent neural network (RNN) approaches demonstrated their quality in the remote sensing field to produce land use mapping using time series of optical images [9] and recognize vegetation cover status using Sentinel-1 radar time series [13]. Motivated by these recent works, we decided to introduce an RNN module to integrate information from the Sentinel-2 time series into our fusion process. In our model we chose the GRU unit (Gated Recurrent Unit) introduced by [11], coupled with an *attention* mechanism [15]. Attention mechanisms are widely used in automatic signal processing (language or 1D signal) and they allow to combine together the information extracted by the GRU model at the different timestamps. The input of a GRU unit is a sequence (x_{t_1},..., x_{t_N}) where a generic element x_{t_i} is a multidimensional vector and t_i refers to the corresponding date in the time series. 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 to combine together these different vectors h_{t_i}, in a single one rnn_{feat}, to better combine the information returned by the GRU unit at each of the different timestamps. The attention formulation we used, from a vector sequence of the learned descriptors (h_{t_1},..., h_{t_N}), is the following one:

$$v_a = tanh(H \cdot W_a + b_a) \tag{1}$$

$$\lambda = SoftMax(v_a \cdot u_a) \tag{2}$$

$$rnn_{feat} = \sum_{i=1}^{N} \lambda_i \cdot h_{t_i} \tag{3}$$

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 (λ_{t_1},..., λ_{t_N}) that allows the contribution of each timestamp to be weighted h_{t_i} through a linear combination. The $SoftMax(\cdot)$ [9] function is used to normalize weights λ so that their sum is equal to 1. The output of the RNN module is the feature vector rnn_{feat}, these features encode temporal information related to ts_i for the pixel i.

[1]Donnes disponibles via http://theia.cnes.fr, pre-treated in surface reflectance via the *MACCS-ATCOR Joint Algorithm* [10] 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).

C. Integration of VHSR information

The VHSR information is integrated in $M^3Fusion$ through a CNN module. Computer vision literature offers several convolutional architectures [16], [17]. Most of these networks are designed to process RGB images (three channels) having size higher then 200x200. Such networks are composed by multiple layers (tens or hundreds). In our scenario, the image patches to analyze have a size of 25x25 and they are described by five channels. In order to propose a CNN module that well fit our scenario and remains computational affordable parameters-wise, we design the CNN module depicted in Figure 2.

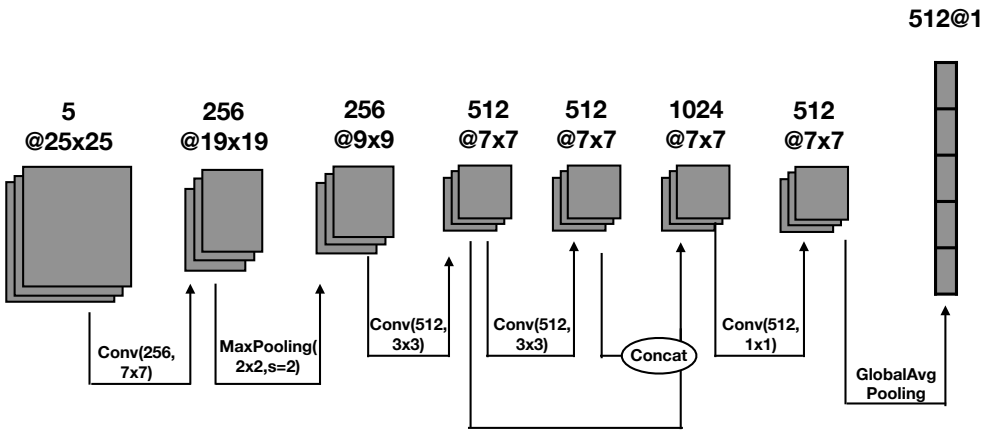


Figure 2: Convolutional Neural Network Structure

Our CNN network applies a first kernel 7×7 to the five-channel patch 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 kernel 3×3 extract 512 features maps each, which are then concatenated and reduced again by a convolution kernel 1×1 (final size 512×7×7). Finally, a Global Average Pooling operation allows to produce a vector of features with size equals to 512.

Each convolution is associated with a linear filter, followed by a Rectifier Linear Unit activation function [18] to introduce non-linearity and a Batch Normalization step [19]. The key points of our proposal are twofold: a) a higher number of filters in the first step and b) the concatenation of features maps at different resolutions. The first point is related to the higher amount of spectral input information (five channels) compared to RGB images. To better exploit the high spectral richness of this data, we have increased the number of feature maps generated at this stage. The second point concerns the concatenation of features maps. With the goal to exploit information at different resolutions we adopt a philosophy similar to [17]. The output of this module is a vector of dimensions 512 (cnn_{feat}) which summarizes the spatial context ($patch_i$) associated to the Sentinel-2 pixel i.

D. Merging descriptors through an End-To-End 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 [7]. This quality is particularly important in a multi-source, multi-scale and multi-temporal fusion process. $M^3Fusion$ leverages this asset to be able to extract complementary descriptors from two sources of information that describe the same pixel from different viewpoints. In order to further strengthen the complementarity and then, the discriminative power of the learned features for each information stream, we adapted the technique proposedin [14] to our problem. In [14], the authors proposed to learn two complementary representations (using two convolutional networks) of 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 through a sum operation. In our case, we have two complementary sources of information (sentinel-2 time series and VHSR data) to which two auxiliary classifiers are attached in order to independently increase their ability to recognize land cover classes. Regarding the classifier that exploits the full set of features, we feed it concatenating the output features of both CNN (cnn_{feat}) and RNN (rnn_{feat}) module together. The learning process will involve optimizing three classifiers at the same time, one specific to rnn_{feat}, a second one related to cnn_{feat} and the third one that consider $[rnn_{feat}, cnn_{feat}]$.

The cost function associated to our model is :

$$L_{total} = \alpha_1 * L_1(rnn_{feat}, W_1, b_1) + \tag{1}$$
$$= \alpha_2 * L_2(cnn_{feat}, W_2, b_2) + \tag{2}$$
$$= L_{fus}([cnn_{feat}, rnn_{feat}], W_3, b_3) \tag{3}$$

where

$$L_i(feat, W_i, b_i) = L_i(Y, SoftMax(feat \cdot W_i + b_i)) \tag{4}$$

Y is 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 which takes as input the set of descriptors returned by a specific module (CNN or RNN) and the parameters to make the prediction (W_1, b_1 or W_2, b_2). $L_{fus}(cnn_{feat}, rnn_{feat}, W_3, b_3)$ is the cost function of the classifier that uses the total set of features ($[cnn_{feat}, rnn_{feat}]$). This last cost function is parameterized through W_3 et b_3. Each of the cost function is modeled through categorical cross entropy, a typical choice for multi-class supervised classification tasks [9].

L_{total} is optimized End-To-End. Once the network has been trained, the prediction is carried out using only 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 [14], operate a kind of regularization that forces, within the network, the extracted features to be discriminative independently.

IV. Experiments

In this section, we present the experimental setting we used and we discuss the results obtained on the data introduced in Section II.

A. Experimental Setting

We compare the performances of the $M^3Fusion$ approach w.r.t the Random Forest classifier (*RF*), which is commonly used for supervised classification in the field of remote sensing [6].

For the *RF* model, we fixed the number of generated random trees at 200 with no depth limits imposed. For the Random Forest method, we used the python implementation supplied by the scikit-learn [20] library. In order to compare these two methods, we supplied the same input data set both to *RF* and $M^3Fusion$ model. Each example of the data set for this competitor has a size of 3 669 which corresponds to $25 \times 25 \times 5$ ($patch_i$) plus 34×16 (ts_i).

In our model we choose the value d (number of hidden units in the recurrent unit GRU) equals to 1 024. We empirically fixed α_1 and α_2 to 0.3. During the learning phase, we used the Adam method [21] to learn the model parameters with a learning rate equal to $2 \cdot 10^{-4}$. The training process is conducted over 400 epochs. The best model regarding the cost function's value is used in the test phase.

We implemented $M^3Fusion$ 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.

In terms of data, we divided the set into two parts, one for learning and the other to test the performances of the supervised classification methods. We used 30% of the objects for the training phase (meaning 97 110 pixels) while the remaining 70% are used for the test phase (meaning 225 638 pixels). We impose that pixels of the same object belong exclusively to the train or to the test set. [5]. The values were normalized in the interval $[0, 1]$ by spectral band.

The assessment of classification performance is done by global precision (*Accuracy*) and *F-Measure* metrics [9].

B. Quantitative Results

Figure 3 shows the results from both classification models in terms of F-Measure per class. We can observe that $M^3Fusion$reaches average better performances compared to the *RF*. The only exception is supplied by class (12) where performance are more than comparable and *RF* obtains slightly better results. Considering classes (1),(3),(4),(5),(7),(8) and (9), which we can consider more difficult to manage since absolute performances are low, the behavior of the deep learning method is always superior to the one exhibited by *RF*.

A class where we can notice a sensible enhancement is the class (10) *Greenhouse crops*. For an *F-Measure* of 0.25 given by *RF*, $M^3Fusion$ achieves an *F-Measure* of 0.58. Indeed, both the appearance and dynamics of the elements in this class are very similar to those of the built-up class. Probably, the spatial information extracted by the deep approach makes possible to derive more discriminative characteristics.

Regarding the Accuracy results, $M^3Fusion$ (resp. *RF*) reach a score rate of 90.67% (resp. 87.39%). For a more detailed analysis, we show in Figure 4a (resp. Figure 4b) the heat map associated with the method's confusion matrix *RF* (resp. $M^3Fusion$). We can observe that the heat map gives a good overview on the behavior of the two methods. First of all, we can observe that the *Random Forest* is noisier, especially on the diagonal. This noise locates the classifier's errors in its decision.

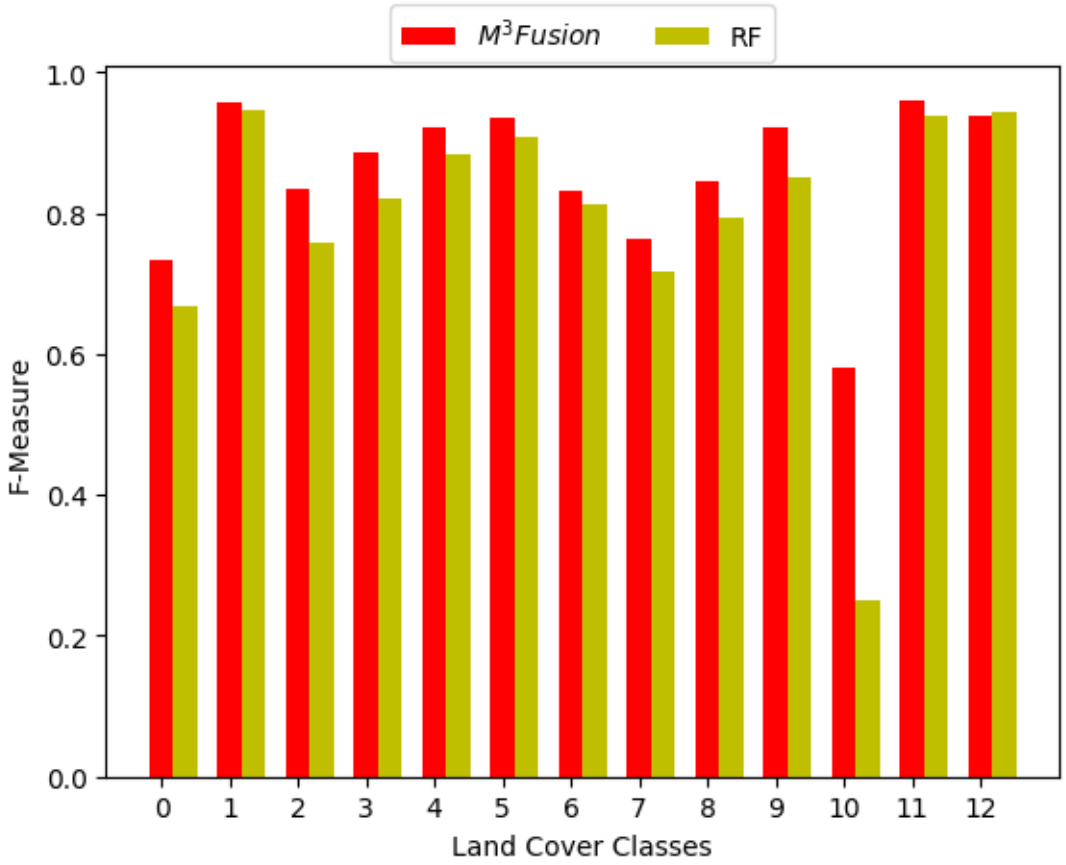


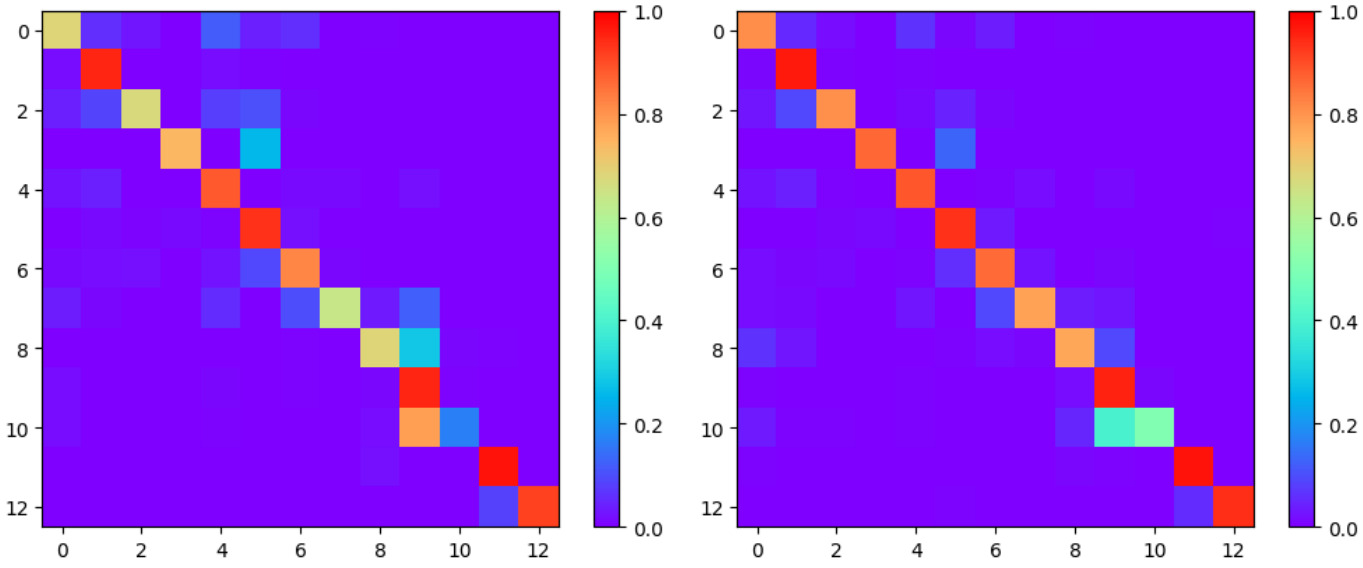
Figure 3: F-Measure by class of the two classification methods.

This behavior is particularly evident in class (10) where the majority of the elements of this class are categorized as class (9). About the $M^3Fusion$ method, we can observe a more coherent structure along the diagonal of the heat map, which indicate less noise. Talking about class (10), also our method tends to make some confusion between classes (9) and (10), but this phenomenon is less evident in relation to the *Random Forest* method since most of the elements are better classified.

The results presented so far are linked to a single split 30%/70% of our data set. It is known that, depending on the split of the data, the performances of the different methods may vary because simpler or more difficult examples can be involved in the train or test set. With the objective of a first understanding of the robustness of our method, regarding this phenomenon, we produced four other splits of the dataset, using the same protocol. The results of the five splits are shown in the Table II. We can observe that the behavior of the two methods, in relation to the different splits, is similar: both methods obtain the best performance on the second split and the worst results on the third split. This behavior is related to the phenomenon we mentioned earlier. On the other hand, we can see that $M^3Fusion$ always gets the best results on all splits with an Accuracy gain (resp. en F-Measure) varying between 2.28 and 4.04 (resp. 2.65 and 4.53). We can highlight two more important facts, $M^3Fusion$ seems to be more stable than the *Random Forest* method. We observe this behavior on split number 3 where the performance of the propositional classifier decreases by more than 3 points compared to its best result. Considering $M^3Fusion$, the difference between the best and worst score is around 2 points. Finally, we can note that for the results relating to the deep learning method, the difference between the Accuracy value and the F-Measure value is minimal, the two values are always rather similar and aligned. On the contrary, for the *Random Forest* method we can see some discrepancy between the two measures. Accuracy measurement is always about half a point higher. Looking closely to the results, we found that *RF* seems to be more influenced by the class imbalance giving more chance in its decision to the majority classes.

C. Qualitative Results

In addition to the numerical evaluations reported in the previous section, we also propose a first qualitative evaluation of the map produced by $M^3Fusion$. The map obtained by the $M^3Fusion$ is also shown in Figure 5 for a qualitative overview. The recognition of the majority classes, i. e. the areas cultivated with sugar cane on the coast, as well as the various degraded



(a) (b)

Figure 4: Confusion matrix of *Random Forest* (a) and $M^3Fusion$ (b).

Essai	*RF*		$M^3Fusion$		*Gain*	
	Acc.	*F-Meas.*	*Acc.*	*F-Meas.*	*Acc.*	*F-Meas.*
1	87.39	87.11	90.67	90.67	+3.28	+3.56
2	88.47	88.05	91.52	91.39	+3.05	+3.34
3	85.21	84.62	89.25	89.15	+4.04	+4.53
4	88.33	88.05	90.61	90.7	+2.28	+2.65
5	87.29	86.88	90.09	89.96	+2.8	+3.08

Table II: Accuracy and F-Measure of the two methods on five different random splits of the data set

natural areas (grasslands, savannas and forests) and the urban fabric, seem to be well localized and regular, with a less important presence of noise than on the map obtained by *Random Forest* (not reported for brevity).

Some comparisons between the two maps are provided at the scale of some remarkable details in Figure 6: in the line above, a fragment of urban areas is displayed, where the presence of noise is particularly marked for the *RF* map (in the middle), as shown by the transition zones between buildings that are often interpreted as market gardening, an effect that does not occur on the $M^3Fusion$map (right). A particular interesting effect concerns the artifacts of the *RF* map due to the presence of clouds or shadows (detail on the bottom line) on the VHSR image, which are not produced with the proposed method: this is probably due to an *RF* bias in favor of information from the VHSR, a situation that does not occur with the proposed approach.

V. Conclusions

In this article, we proposed a new deep learning architecture for the fusion of satellite data with high temporal/spatial resolution and very high spatial resolution to perform land use mapping. Experiments carried out on a real study site validate the quality and the results of our approach compared to the ones obtained by a common machine learning methods usually employed in the field of remote sensing for the same task. In the future, we plan to study the extension of our architecture to take into account other complementary data sources.

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 and the GEOSUD project with reference ANR-10-EQPX-20, as well as from the financial contribution from the Ministry of Agriculture's "Agricultural and Rural Development" trust account. This work also used an image acquired under the CNES Kalideos scheme (La Réunion site).

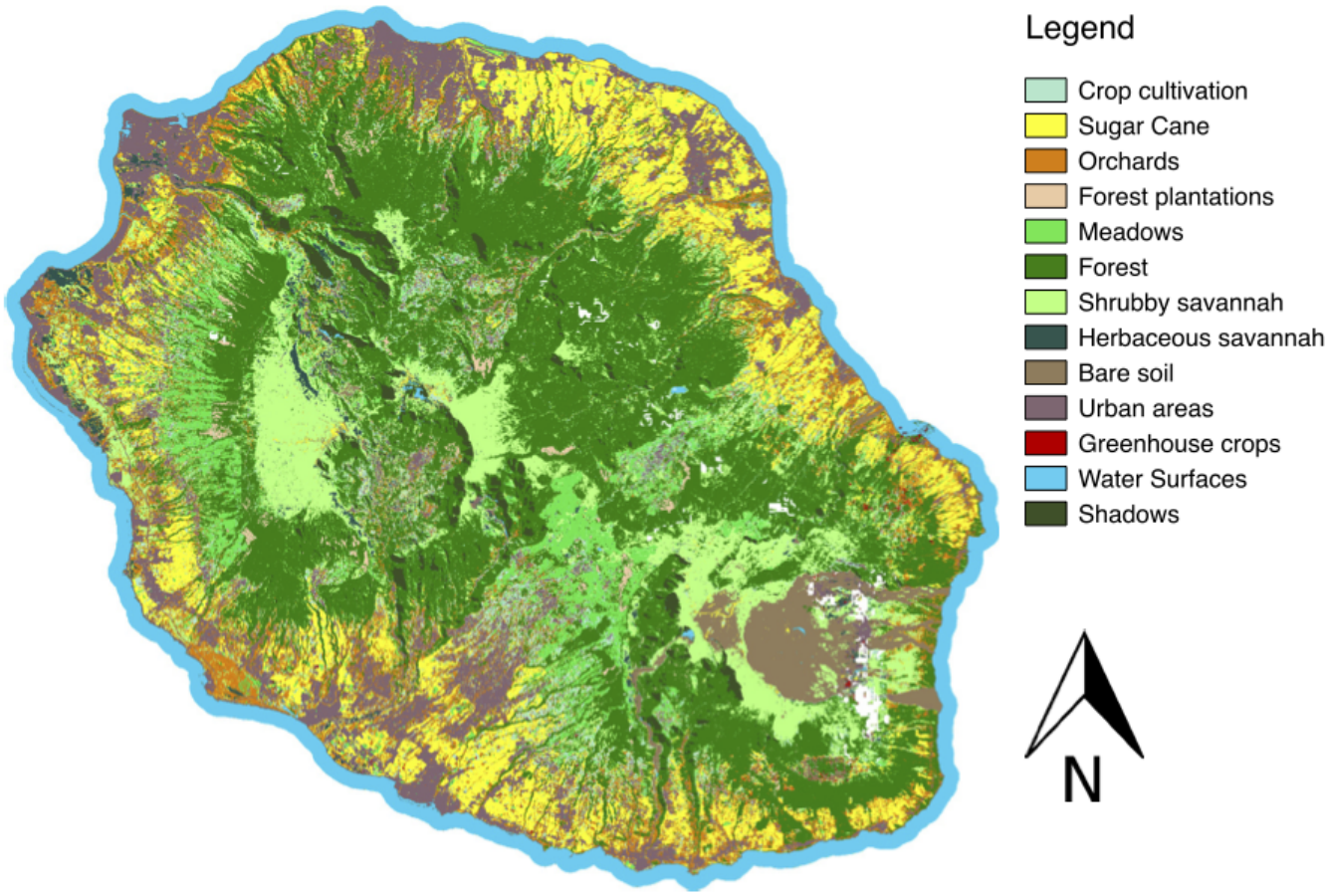


Figure 5: Map produced by $M^3Fusion$

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] 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.

[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] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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, VENμS and Sentinel-2 Images," *Remote Sensing*, vol. 7, no. 3, pp. 2668–2691, 2015.

[11] 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.

[12] 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.

[13] 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.

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

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

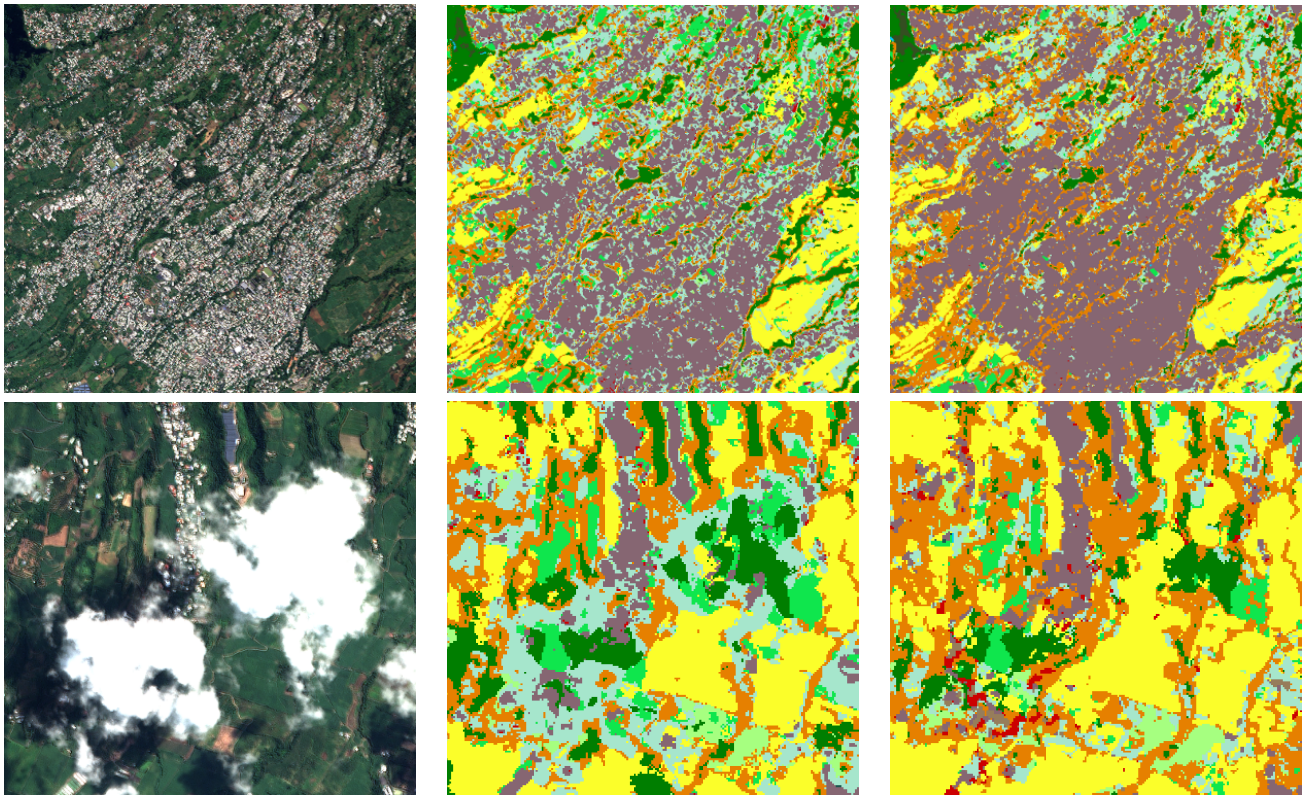


Figure 6: Classification results obtained with *Random Forest* and $M^3Fusion$: right to left, excerpt from SPOT6/ image7, classification by ***RF***, classification by $M^3Fusion$.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.
[17] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017, pp. 2261–2269.
[18] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML10*, 2010, pp. 807–814.
[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.
[20] 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.
[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.