International Journal of *Geo-Information*

Article

Sentinel 2 Time Series Analysis with 3D Feature Pyramid Network and Time Domain Class Activation Intervals for Crop Mapping

Ignazio Gallo [1,*,†], Riccardo La Grassa [1,†], Nicola Landro [1,†] and Mirco Boschetti [2,†]

1 Department of Theoretical and Applied Sciences, University of Insubria, Via O. Rossi, 9, 21100 Varese, Italy; rlagrassa@uninsubria.it (R.L.G.); nlandro@uninsubria.it (N.L.)
2 IREA CNR, Via Corti, 12, 20133 Milano, Italy; boschetti.m@irea.cnr.it
* Correspondence: ignazio.gallo@uninsubria.it
† These authors contributed equally to this work.

Abstract: In this paper, we provide an innovative contribution in the research domain dedicated to crop mapping by exploiting the of Sentinel-2 satellite images time series, with the specific aim to extract information on "where and when" crops are grown. The final goal is to set up a workflow able to reliably identify (classify) the different crops that are grown in a given area by exploiting an end-to-end (3+2)D convolutional neural network (CNN) for semantic segmentation. The method also has the ambition to provide information, at pixel level, regarding the period in which a given crop is cultivated during the season. To this end, we propose a solution called Class Activation Interval (CAI) which allows us to interpret, for each pixel, the reasoning made by CNN in the classification determining in which time interval, of the input time series, the class is likely to be present or not. Our experiments, using a public domain dataset, show that the approach is able to accurately detect crop classes with an overall accuracy of about 93% and that the network can detect discriminatory time intervals in which crop is cultivated. These results have twofold importance: (i) demonstrate the ability of the network to correctly interpret the investigated physical process (i.e., bare soil condition, plant growth, senescence and harvesting according to specific cultivated variety) and (ii) provide further information to the end-user (e.g., the presence of crops and its temporal dynamics).

Keywords: convolutional neural networks; Sentinel-2 satellite images; 3D feature pyramid network; time domain class activation intervals; time series

check for updates

Citation: Gallo, I.; La Grassa, R.; Landro, N.; Boschetti, M. Sentinel 2 Time Series Analysis with 3D Feature Pyramid Network and Time Domain Class Activation Intervals for Crop Mapping. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 483. https://doi.org/10.3390/ijgi10070483

Academic Editor: Wolfgang Kainz

Received: 13 May 2021
Accepted: 8 July 2021
Published: 13 July 2021

1. Introduction

Producing more (food) with less (consumption of natural resources) is one of the biggest challenges our society is facing to guarantee food security globally and it is, therefore, one of the priorities of UN SDG goals. To achieve this, agricultural production must be supported by sustainable planning and one of the first pieces of information needed is "where and when" crops are cultivated. Satellite Remote Sensing is the best candidate as an information source to perform crop monitoring worldwide, but powerful and robust algorithm to provide temporal and spatially explicit information on crop presence are needed. Sentinel-2 (S2) is a Copernicus Earth observation mission that systematically acquires high spatial resolution optical images of Earth. This mission introduces a paradigm shift in the quality and quantity of open access data, opening a new era for land monitoring systems especially for the agricultural sector. S2 provides multi-spectral data with a spatial resolution from 10 m up to 60 m able to cover 290 km per acquisition with a revising time of 5 days thanks the two satellites constellation. This, however, introduces the notion of big data and therefore we need models that have the ability to exploit this enormous amount of information. In this framework, we want to contribute to research innovation devoted to perform crop identification from the analysis of time series of Sentinel-2 satellite imagery. The goal is to identify an approach that is able to reliably identify (classify) the different crops that are growing in a given area using an end-to-end (3+2)D convolutional neural network (CNN) for semantic segmentation, the method also has the ambition to predict the period in which a given crop is actually growing in a given season.We develop a model based on Features Pyramid Networks (FPN), adapted to process the time series with 3D kernels of a small area associated with each input sample, and which is able to provide as output a segmentation map using 2D kernels. Furthermore, we investigate and propose a solution to understand how CNN identify the time intervals that contribute to the determination of the output class-Class Activation Interval (CAI). This solution allows us to interpret the reasoning made by CNN in the classification of a single pixel. We demonstrate in a variety of experiments that our network is able to identify discriminatory time intervals in the input features domain despite the CNN has been trained only to solve a classification task (i.e., spatial solution). Therefore, with our CAI method we are able to provide information on "when" the class associated with a pixel is present in the time series of Earth Observation (EO) data. CAI method becomes useful to discriminate when crops are the only one cultivated (e.g., summer maize) or represent the second crop of the season (e.g., winter wheat followed by maize). In the latter case, maize is generally sown later in the season following the grain harvest and soil preparation. The ability to provide such information will help characterize crop systems (single or double crops) alongside the class of simple crops. Thanks to CAI, the information on the sowing period will provide an idea of the cultivated variety and the destination of crops (e.g., corn for silage or forage).

Moreover, the proposed approach has a two-fold importance: demonstrate the capacity of the network to correctly interpret the physical process investigated and provide additional information to the end-user (i.e., crop presence and its temporal dynamics. The provision of CAI output is a way to assess robustness of model for each semantic class because provide an explicit representation of the time period in which crop is likely to be cultivated, such information for a specific study area can be confirmed by expert knowledge or provide added value information for final user. Figure 1 provides a graphical representation of CAI information for a spatial subset of the analysed S2 data where maize is cultivated. Low CAI values occur at the beginning of time series (December to May) when crop residue (other crop) are present while the indicator values significantly increase in the proper growing period (May to September).

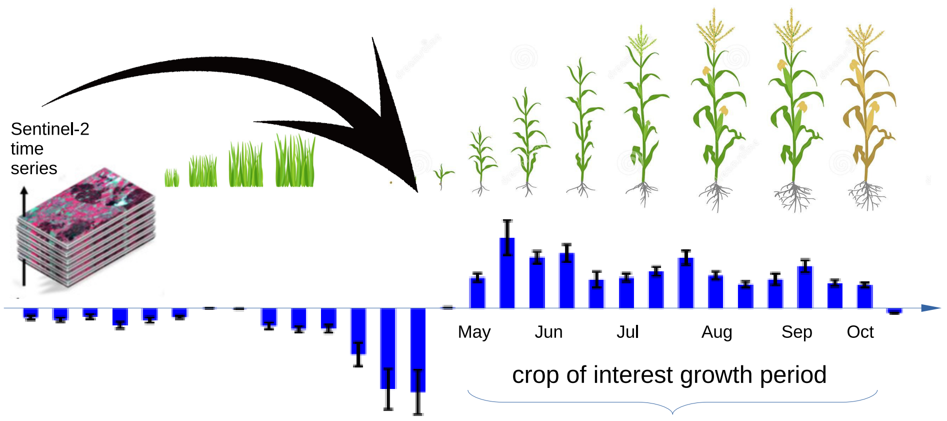

Figure 1. Representation of proposed technique for class activation interval (CAI). The fully convolutive CNN trained for image segmentation is able to identify time intervals of the input time series that determines class presence. The CAI can be computed in a single forward step: for the class maize results show data relevance from May to October.

There are several types of convolutional neural networks (CNNs), and all of them can greatly help improve the speed and accuracy of many computer vision tasks. In particular, CNN's 3D models are often used to improve object identification in videos or 3D volumes, such as security camera videos [1] and medical scans of cancerous tissue [2].

The spatiotemporal dimension of Sentinel-2 satellite imagery has many similarities to video; for this reason, we adopt models similar to those used for analysis of videos. In recent years, spatiotemporal data has been addressed through CNN models that follow three main ideas: CNN 2D (e.g., Two-stream ConvNets [3] and Temporal Segment Network (TSN) [4]), CNN 3D (e.g., SFSeg [5] and 3D ResNet [6]) and (2+1)D CNN (e.g., P3D [7] and R(2+1)D [8]). Our proposal partly follows the idea of 3D CNNs used for video data sets, but at the same time uses 2D kernels (i) to create segmentation maps and (ii) to predict the activation intervals of classes in the time domain

Recently, the performance levels of 3D CNNs in various fields have greatly improved [6] and in addition we have a huge amount of free satellite data [9] that can be easily interpreted by 3D models. Motivated by the success of the [10] 2D-FPN network used for multi-class semantic segmentation, successfully applied also on satellite data for semantic segmentation starting from RGB images [11], in this paper we develop a FPN (3+2)D for multi-class semantic segmentation, based on 3D and 2D convolution layers. In particular the model we proposed was designed for crops semantic segmentation to improve automatic crop recognition accuracy by implementing 3D multi-scale capabilities and to increase the spatial and temporal resolution of thematic product. This model is designed for crops semantic segmentation to improve crops recognition accuracy by implementing 3D multi-scale capabilities and to increase the spatial and temporal resolution of crops.

The Feature Pyramid Network (FPN) looks a lot like a U-shaped Convolutional Neural Network (U-Net) [12]. Like the U-Net, the FPN has a lateral connection between the bottom-up pyramid and the top-down pyramid. However, where U-net simply copies the features and adds them, FPN applies a 1×1 level of convolution before adding them. This allows the bottom-up pyramid called the "backbone" to be pretty much anything we want. Due to this greater flexibility of the FPN, we have chosen it and adapted it to our particular (3+2)D problem. U-Net has also been adapted to 3D segmentation problems and has been successfully applied to many 2D and 3D segmentation problems [13]. Unfortunately, we cannot compare directly with the U-Net because we would have to modify it in order to to be able to work on a 3D input and a 2D output at the same time, but knowing that FPN and U-Net are two very similar models and that FPN is more flexible compared to U-Net, we have decided to work only with the FPN.

Recents papers by Zhou et al. [14,15] have shown that the convolutional units of various layers of convolutional neural networks (CNNs) actually behave as object detectors despite receiving no supervision over the location of the object. A similar paper [16] proposes a technique for making CNNs more transparent by adding "visual explanations" to a large class of models. They demonstrated that the network can maintain its remarkable object localization capability until the final layer, thus allowing it to easily identify discriminating image regions in a single forward pass for a wide variety of activities, even those for which the network was not originally trained. Inspired by the Class Activation Maps (CAM) proposed by Zhou et al. [14], we propose to expand the proposed (3+2)D FPN CNN with a mechanism that allows visualizing the time interval of the time series that contribute to the determination of the class for each pixel.

With the (3+2)D FPN we want to be more precise in answering the question "*where a specific crop is located*" inside an image, but we do not know how to answer the question "*when that crop was present*" inside a time series. For this reason we have added to the network a mechanism to predict when the class is present in the input time series without needing to supply more ground-truth to the network. In a similar way to what happens in [14], where the network is adapted to understand "*which area of the input image*" contributes most to the determination of the output class, we have proposed a solution to predict for each pixel "*in which time interval*" a specific class is present in the input time series dataset. Exactly as it happens for the CAM, where we can say which area of the image contributed to determining the output class and say nothing about the other classes, we use the CAI to tell when the class was active in the input time series without saying anything about the other classes on that pixel. This is to say that our approach is multi-class on every pixel that the model sees as input, but it is not multi-label.

The novelties we propose in this paper are listed below:

- We propose a new semantic segmentation model suitable for remote sensing time series.
- We add to the proposed CNN a mechanism that allows visualizing the time interval of the time series that contribute to the determination of the class for each pixel.
- We exceed by about 4% the state of the art on a public dataset with satellite time series.

2. Proposed Method

The method we propose in this paper is composed of two main innovative parts, the Convolutive Neural Network that uses 3D convolution on time series and the Class Activiation Interval used to understand when a class (associated with a single output pixel) is active with respect to the time series analyzed in input.

2.1. (3+2)D Features Pyramid Network

Recent advances in computer vision suggest that Features Pyramid Networks (FPNs) [10] are very effective in detecting objects at different scales, but they are also a great tool for many image segmentation problems [17,18]. However, traditional FPNs are designed for 2D images. Here, we propose an FPN network that uses 3D kernels on the input time series and on almost the entire network, except on the last layers near the output where we use 2D kernels to be able to segment a map.

Figure 2 shows a graphical representation of the proposed end-to-end model. The leftmost column represents the path from the bottom (input data) to top, i.e., the feed-forward computation of the backbone ConvNet. For this bottom-up pathway, a known model is used and in Figure 2, we used a modified version of the ResNet 50, but we conducted experiments with others models. Unlike [10] where the network is designed for large images, we change the model in some places to be able to work with small image in general (i.e., 48×48 pixels size in the specific case). For example, as input, we use a kernel size equals to $(7, 3, 3)$, with stride $= (1, 1, 1)$ and padding $= (3, 1, 1)$, to avoid arriving at the blocks *c*4 or *c*5 (see Figure 2) with an image size $= 1 \times 1$. We removed the smooth layers after the blocks *p*4, *p*3 and *p*2 (see Figure 2) to avoid destroying the signal too much since we use very small images.

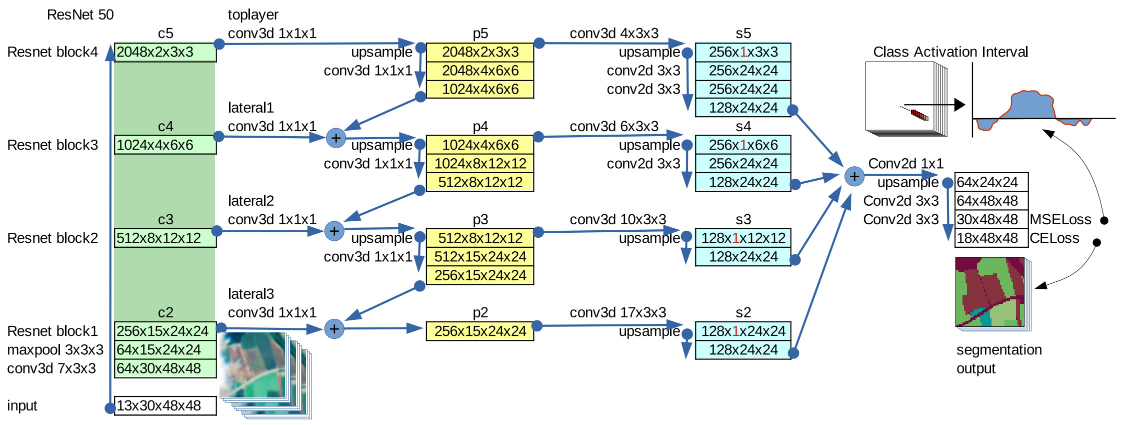

Figure 2. Summary scheme of the proposed convolutional model. The green layers on the left represent the bottom-up pathway, here represented by a modified version of the ResNet 50. The output of the last layer of each block is used to enrich the top-down path (yellow layers) by lateral connection. The feature maps from the bottom-up pathway and the top-down pathway are merged by element-wise addition. This set of feature maps is called $\{P2, P3, P4, P5\}$, corresponding to $\{C2, C3, C4, C5\}$ that are, respectively, of the same shapes. To obtain the last block for prediction $\{S2, S3, S4, S5\}$, some 3D kernels are used which lead to features maps with a dimension of size 1 that will be removed. The outputs of the last two 2D convolutions are used by the MSE Loss and the Cross-Entropy Loss to calculate the segmentation map and the class activation intervals.

Let $I \in \mathbb{R}^{B\times T\times H\times W}$ be the time series having T images $H \times W$ with B channels, and let $\hat{y} = f(I)$ be the part of the network that transforms the image I into the NDVI index [19] (see Equation (2)) associated with each pixel. That is, $\hat{y} \in \mathbb{R}^{T\times H\times W}$ is the output of the second last layer called NDVI layer. The final output $y_{i,j,c}$ of the model, calculated for each pixel (i, j) of the segmentation map and for each class $c \in C$, can be written with the following convolution operation between a kernel $T \times K \times K$ and the $\hat{y}$ values:

$$y_{i',j',c} = \sum_{t=1}^{T}\sum_{j=1}^{K}\sum_{i=1}^{K} w^{c}_{i,j,t} \cdot \hat{y}_{i'+i,j'+j,t} \tag{1}$$

Our bottom-up and top-down blocks are of the same size to avoid losing too much information from time series. Furthermore, in the semantic block before the output, we switch from 3D kernels to 2D kernels. Finally, in the NDVI layer we use an MSE loss function to predict the following Normalized Difference Vegetation Index (NDVI)

$$NDVI = \frac{NIR - RED}{NIR + RED} = \frac{B8 - B4}{B8 + B4} \tag{2}$$

where $B8$ and $B4$ are two spectral bands of the Sentinel-2 satellite in which each pixel corresponds to a geographical area of 10×10 m. NDVI is a radiometric measure of the photosynthetically active radiation absorbed by chlorophyll in green leaves and is, therefore, a simple indicator for assessing the presence and quantity of green vegetation in the observed target. In agriculture monitoring, time series of NDVI allows to follow the planting period, plant growing and harvesting of crops. Thanks to this regression before the classification layer, the network must say which NDVI values of the input time series contribute to the determination of the class activation. We exploit this dependence to be able to calculate the Class Activation Interval (CAI) described in the following section.

In Figure 3, we show two examples of what the NDVI layer sees during the training phase as target and what it predicts in each pixel (shown as the mean and variance of the values of all the pixels belonging to that class). To produce the target values we calculate the NDVI (Equation (2)) for each pixel having the same target class and then we aggregate all these indices into a single vector $t_{i,j} = [t_1, t_2, \ldots, t_T]$ so that all pixels $(i, j) \in c$ have associated the same vector. To aggregate all NDVI values of a class we compute the average plus a moving maximum 1D filter of size 5 and the output is filtered by a ReLU function to obtain only positive values. Looking at the example of red dashed curve of the average NDVI signal in Figure 4 we can observe the period of growth and the harvest made after the summer and then, the harvest, something else started to grow. From the same figure we can observe that real NDVI values are very noisy and events such as the passing of a cloud can lead to classification errors. For this reason we have decided to replace real NDVI values with our computed average NDVI.

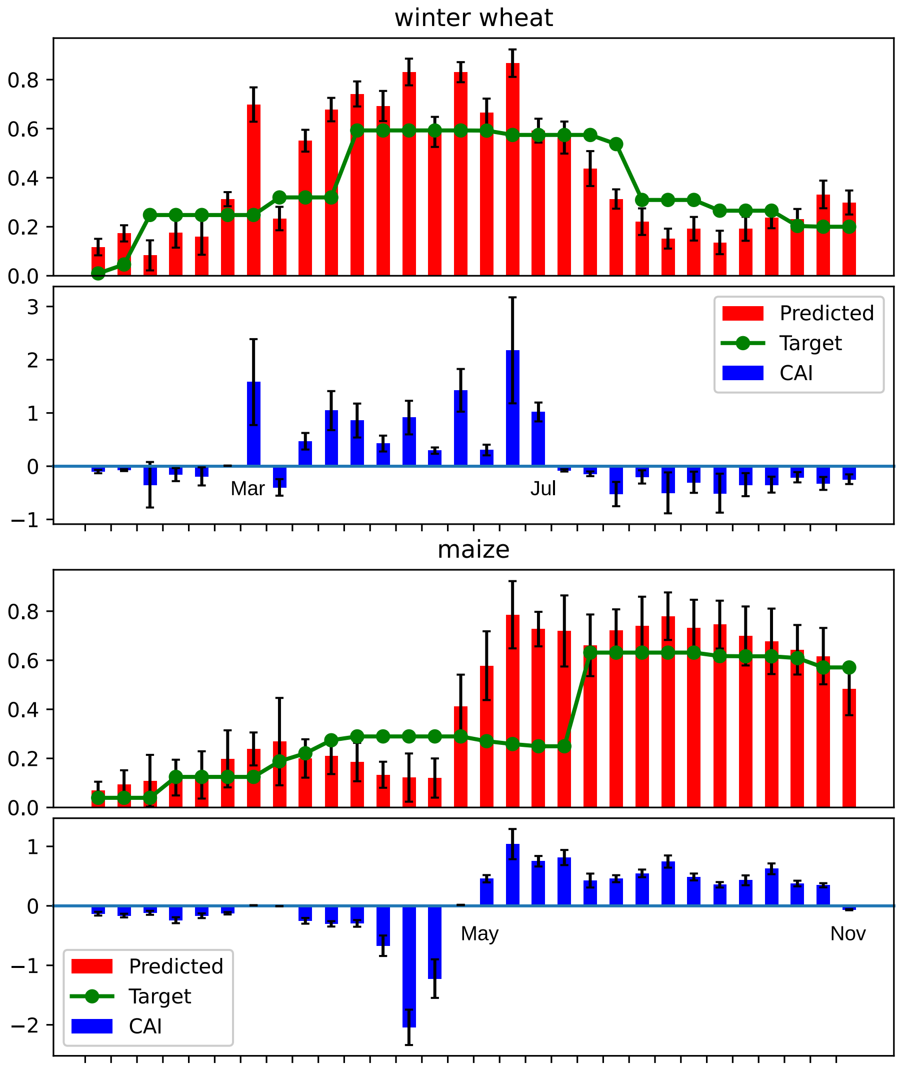

Figure 3. Two real examples showing the target NDVI index (green line) for two different test examples extracted from the Munich test set. The outputs predicted by the network for each pixel of the input sample are represented by mean and variance values (red bars). In blue, the value of the Class Activation Interval for the two classes *winter wheat* (**Top panel**) and *maize* (**Bottom panel**).

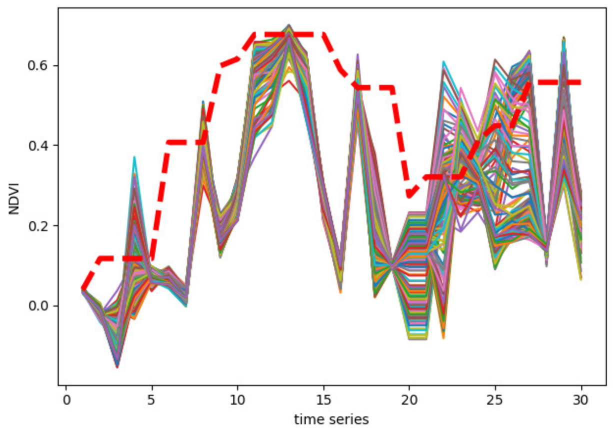

Figure 4. The continuous colored curves are the real NDVI of all the "winter wheat" class pixels available in a patch of size 48 × 48 pixels, in one year (30 samples). While the red dashed curve represents the average value computed by us. Note that the average value is always positive because we filtered the signal with a RELU function.

We used two loss functions to train the model, an MSE loss to learn the trend of the NDVI and a Cross Entropy loss to learn the class labels associated with each pixel. So we can introduce the first loss for a point (i, j) as:

$$\mathcal{L}_{i,j}^{MSE}(\hat{y}, t) = (\hat{y}_{i,j} - t_{i,j})^2 \tag{3}$$

For the last layer, we used a cross-entropy loss function which can be written as follows

$$\mathcal{L}_{i,j}^{CE}(y, c) = \alpha_c \left(-log \left(\frac{e^{y_{i,j,c}}}{\sum_{k=1}^{C} e^{y_{i,j,k}}} \right) \right) \tag{4}$$

where $y \in \mathbb{R}^{C \times H \times W}$ and α_c is a weight for the class c used to balance the training set. The loss function of one sample is obtained by summing all the two losses described before:

$$\mathcal{L} = \sum_{j=1}^{H} \sum_{i=1}^{W} \mathcal{L}_{i,j}^{CE}(y, c) + \lambda \sum_{j=1}^{H} \sum_{i=1}^{W} \mathcal{L}_{i,j}^{MSE}(\hat{y}, t) \tag{5}$$

Here, λ is a scalar used as a regularization hyperparameter, whose value can be optimized for better results. Note that Equation (5) will include a new index for the minibatch and then will be reduced using the sum.

2.2. Class Activation Intervals

In this section, we describe the procedure for generating class activation interval (CAIs) using a (3+2)D fully-convolutional neural network. A CAI for a particular category indicates the discriminating intervals used by CNN to identify that category (see some examples in Figures 1–3 and 5). The procedure for locating these time intervals is illustrated in Figure 5. As shown in Figure 2, two convolutional layers with filters 2D of size $K \times K = 3 \times 3$ were used to obtain the last two layers to which the two loss of Equations (3) and (4) are associated. Suppose we have a problem with C classes and therefore in correspondence of each pixel (i', j') in input, in output we have a vector of elements $[y_1, y_2, \ldots, y_C]$. As described in the following Equation (6), each of these elements y_c is obtained as the output of a convolution, and therefore depends only on the elements contained in a neighborhood $(i', j', 0) \ldots (i' + K, j' + K, T)$ of the previous layer. More formally

$$y_{i',j',c} = \sum_{t=1}^{T} \sum_{j=1}^{K} \sum_{i=1}^{K} w_{i,j,t}^{c} \cdot \hat{y}_{i+i',j+j',t} = \sum_{t=1}^{T} n_{i',j',t}^{c} \tag{6}$$

Analyzing the Equation (6) it is clear that, once the label c of the winner class has been obtained, it is possible to go back to understand on which time interval this output value depends. So, to get the Class Activation Intervals in a point (i', j'), that is $CAI_{i',j'} = [n_{i',j',1}^{c}, \ldots, n_{i',j',t}^{c}]$ we have to do as follows

$$n_{i',j',t}^{c} = \sum_{j=1}^{K} \sum_{i=1}^{K} w_{i,j,t}^{c} \cdot \hat{y}_{i+i',j+j',t} \tag{7}$$

Positive $n_{i',j',t}^{c}$ values indicate that the date t of the time series contains information useful for determining the class, while negative values indicate that the date t does not contribute to the determination of the class.

Note that the loss of Equation (5) can be computed in parallel, but the NDVI layer must be placed before the output segmentation layer if we want each output class label to depend on NDVI values. If we put the two layers in parallel, we cannot say anything about the dependence between class label and NDVI activation with respect to the input time series and therefore we cannot calculate the CAI.

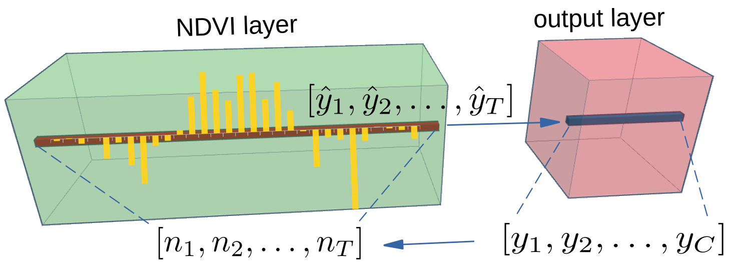

Figure 5. Schematic representation of the last two layers of the neural model involved in the computation of the Class Activation Intervals. Starting from the aggregate features $[\hat{y}_1, \dots, \hat{y}_T]$ it is possible to obtain the vector of activations $[y_1, \dots, y_C]$ to determine the class associated with a pixel and vice versa, it is possible to obtain the Class Activation Intervals $[n_1, \dots, n_T]$ starting from the values $[y_1, \dots, y_C]$.

3. Dataset

In our experiments we used the Munich dataset used in [20] which contains squared blocks of 48 $\times$ 48 pixels including 13 Sentinel-2 bands (see some samples in Figure 6). Each 480-m block was mined from a large geographical area of interest (102 km × 42 km) located north of Munich, Germany. In our experiments we used the split 0 of the dataset, containing 6534 blocks for the training set, 2016 blocks for the test set and 1944 blocks for the evaluation set.

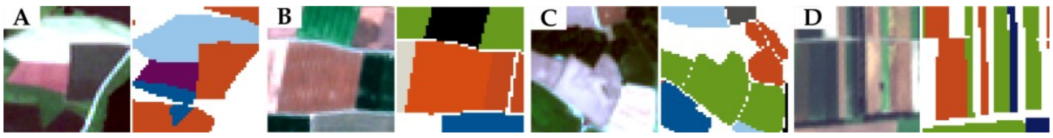

Figure 6. (**A**)–(**D**) Four random samples of input-output pairs from the Munich dataset. The input on the left is shown as an RGB image, while the output shows the class labels as colors.

The ground truth is a 2D image containing the segmentation of the various crops present in each sample, where each pixel has an associated class label obtained from the two growing seasons 2016 and 2017. The segmentation data are not for each date of the time series but are associated with an entire year, so in each pixel, the label represents the harvest crop declared in that year. The 17 classes in the dataset are reported in Table 1. The dataset does not contain information on when a crop is present (sowing and plant growth) and when it is absent (harvest) during a year of observations. The dataset was divided into training, validation and test sets. The dataset is very unbalanced and the cardinality of the last two sets is shown in Table 1. The cardinality reported in the paper [20] does not correspond to ours despite having used the same splits, probably because we used a time series of different number of samples (we extracted 30 samples in each time series) or because some image augmentation was used in their paper. The original dataset can be downloaded from [21].

Table 1. Numerical results of the comparison made with what was published in paper of Rußwurm and Korner. Our model uses a Resnet101, trained for 300 epochs on the training set. We report the comparison measures on the test set and on the validation set (we did not do hyperparameter tuning). In this experiment no techniques or weights were used to solve the problem of unbalanced classes. All measurements, except the Kappa, are in percentages.

	[20]				Our (Eval Set)				Our (Test Set)			
	P	**R**	**F1**	**#pix.**	**P**	**R**	**F1**	**#pix.**	**P**	**R**	**F1**	**#pix.**
sugar beet	91.90	78.05	84.40	59k	90.95	**93.66**	92.29	19k	**96.37**	89.65	**92.88**	33k
oat	74.95	65.30	69.55	36k	79.17	68.13	73.23	23k	**83.28**	**69.88**	**75.99**	30k
meadow	89.45	85.35	87.35	233k	89.51	87.45	88.47	149k	**93.27**	**89.53**	**91.36**	167k
rapeseed	95.80	92.95	94.35	125k	95.98	97.64	96.81	105k	**95.99**	**97.99**	**96.98**	92k
hop	94.45	81.10	87.20	51k	95.12	93.26	94.18	71k	**95.55**	**93.94**	**94.74**	39k
spelt	65.20	**63.90**	61.60	38k	66.39	51.12	57.76	17k	**68.02**	59.69	**63.58**	23k
triticale	**65.90**	**56.45**	**60.75**	65k	57.24	36.03	44.22	34k	58.20	44.59	50.49	41k
beans	92.60	75.15	82.40	27k	91.40	79.03	84.77	15k	**95.26**	**88.63**	**91.82**	19k
peas	77.05	56.10	64.85	9k	**87.07**	65.26	74.61	9k	81.21	**83.77**	**82.47**	6k
potato	**93.05**	81.00	86.30	126k	91.37	91.58	91.48	74k	90.94	**92.32**	**91.63**	84k
soybeans	86.80	79.75	82.75	21k	96.05	**84.33**	**89.81**	12k	**96.96**	81.82	88.75	14k
asparagus	85.40	78.15	81.60	20k	70.86	**92.34**	80.19	1k	**95.72**	84.98	**90.03**	14k
wheat	88.90	94.05	91.40	806k	**93.85**	**96.68**	**95.24**	582k	92.66	95.74	94.17	531k
winter barley	**93.85**	89.75	91.70	258k	93.52	95.01	**94.26**	214k	93.19	**95.28**	94.23	170k
rye	81.15	54.45	64.60	43k	**83.58**	57.51	68.14	15k	82.36	**59.14**	**68.85**	25k
summer barley	82.70	85.95	84.15	73k	**84.94**	87.71	86.30	52k	84.60	**89.54**	**87.00**	52k
maize	91.95	96.55	94.20	919k	96.74	**98.03**	**97.38**	713k	96.44	**98.19**	97.31	604k
weighted avg.	89.70	89.60	89.40		**93.22**	**93.55**	**93.39**		92.94	92.68	92.81	
			Overall Accuracy	89.60			Overall Accuracy	**93.55**			Overall Accuracy	92.94
			Overall Kappa	0.87			Overall Kappa	**0.92**			Overall Kappa	0.91

4. Experiments

We have conducted three main groups of experiments: initially, in Section 4.1, we test some techniques to neutralize the effect of unbalanced datasets. As a second experiment, in Section 4.2, we compare the proposed model with the literature. Finally, in Section 4.3, we analyze the CAI produced by the trained model.

In our experiments, we used some well-known metrics to evaluate the goodness of our model. In particular, we mainly use the overall accuracy, Kappa [22], Recall, Precision, and F-measure coefficients to compare our results with the results published in [20].

We have not conducted any systematic experiments to understand what could be the best value to assign to the hyper-parameter λ for the loss function defined in Equation (5) and therefore we set this value to $\lambda = 1$. We run each experiment for 300 epochs. The optimizer is SGD [23], with momentum equals to 0.9, weight decay 0.001 and with an initial learning rate of 0.01 and a scheduler that uses a cosine function to reduce the learning rate after each epoch. All the time series used are made up of 30 randomly extracted samples from each of the two available years.

The trained models are available on pythorch-hub [24] and the source code to run the experiments is available on a gitlab repository [24].

4.1. Class Imbalance Experiments

Almost all land use segmentation datasets have the problem of the predominance of some classes over others. Which classes predominate with respect to the others, this depends on the geographical area, the season, and the extent of the territory analyzed. Furthermore, for this reason, multi-class segmentation for land cover problems through the analysis of satellite images still remains a challenging problem. The number of samples from the training dataset used to estimate the error gradient during training is called the batch size and is an important hyperparameter that affects the resulting trained model. If the dataset is unbalanced, the number of pixels belonging to each class that the neural network uses to compute the gradient depends on the batch size. If the batch is too large then the predominant classes completely crush the classes that have very few samples, while if the batch is small then the number of pixels for each class is more balanced. For this reason, in this first experiment, we experimentally analyze the effect of batch size together with two class weighing techniques.

In this section, we analyze the performance of the proposed model, without the NDVI layer, using different techniques to counter the effect due to the unbalanced dataset. In particular, we use the two different types of α_c weights of the loss function (see Equation (4)) described below, to try to counter the effect of the unbalancing of the classes we have in the dataset, and then we compare it with a weight $\alpha_c = 1$ for all classes (no weights). As bottom-up pathway of the proposed (3+2)D FPN, in these experiments we use a ResNet1 01.

In Table 2, we have some results from the model which uses only the cross-entropy defined in Equation (4) but without the α_c weight (no weights) and we compare it with the weighting scheme (batch weights) which is based on "Effective Number of samples" within each batch [25] assigned to the α_c weight, and with weighting strategies based on the total number of samples present in each class (global weights) even assigned to the α_c weight. The weighting scheme proposed in [25], counts for each batch of the training set, the number of actual samples n_c in each class c. The effective weight used in each batch is defined using a simple formula $\alpha_c = (1-\beta)/(1-\beta^{n_c})$, where $\beta \in [0,1)$ is a hyper-parameter. In *global weights*, we use a weight $\alpha_c = \max_c(n_c)/\sqrt{n_c})$ for each class, computed using the entire training set.

Table 2. Results on Munich validation set to test the effects of the batch size training parameter and three different ways to weight the loss function. Overall accuracy (OA), Kappa and the weighted average measures Recall (R), Precision (P) and F-measure are used to compare two different techniques to handle unbalanced datasets while the batch size is varied between 2 and 32.

Weight	Batch	OA	Kappa	w. R	w. P	w. F1
batch	2	91.98	**0.920**	91.98	91.78	91.82
batch	4	91.64	0.894	91.64	91.18	91.30
batch	8	90.55	0.879	90.55	89.84	90.00
batch	16	89.39	0.865	89.39	89.09	89.14
batch	32	85.82	0.820	85.82	85.16	85.31
global	2	93.17	0.913	93.17	92.89	92.96
global	4	92.10	0.899	92.10	91.54	91.62
global	8	91.06	0.886	91.06	90.40	90.43
global	16	90.34	0.877	90.34	89.86	89.97
global	32	87.22	0.837	87.22	86.48	86.59
no	2	**93.71**	**0.920**	**93.71**	**93.41**	**93.56**
no	4	92.33	0.902	92.33	91.71	91.73
no	8	91.21	0.887	91.21	90.35	90.44
no	16	90.54	0.879	90.54	90.00	90.07
no	32	87.56	0.840	87.56	86.69	86.54

Analyzing the overall accuracy (OA), Kappa, weighted Recall (w.R), Precision (w.P) and F-measure (w.F1) in Table 2 and the graphs in Figure 7, we can see that loss function weighting techniques do not lead to any advantage on this dataset. On the other hand, it should be noted that the batch size (see the second column labeled *batch* in Table 2) has a great influence on the result and then a very small batch size allows to obtain the best results. From Table 1) we see that the class that has the lowest number of pixels (column #pix) on the validation set is the *asparagus* class. While in Figure 7 the effect of the weight α_c of the loss function and the effect of the batch size can be analyzed as a function of the number of samples for each class. Analyzing, for example, the *asparagus* class when the weight $\alpha_c = 1$ for all the classes (plot on the top of Figure 7) we can see that the best results are obtained when we use a small bach size. As a consequence of these experiments, in the other experiments we do not use the weight α_c.

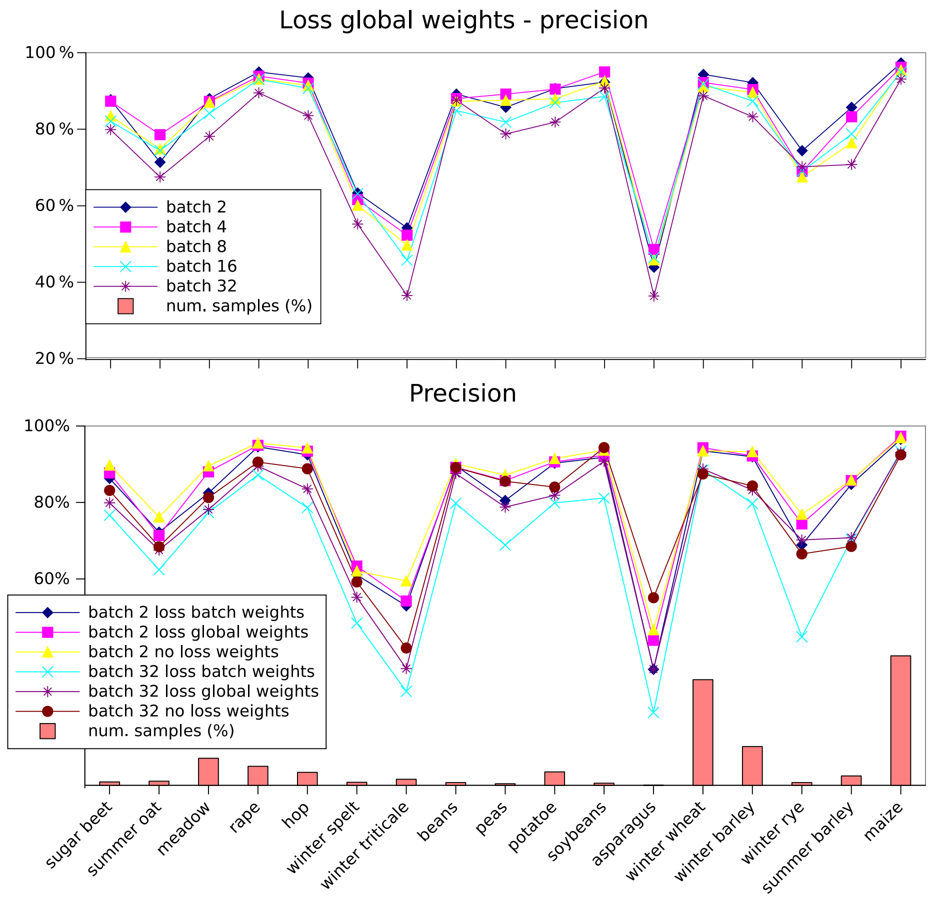

Figure 7. Precision of the FPN 3D model as the batch size varies on the dataset Munich. The plot on the top shows how the precision for each class changes as the batch size in the training phase changes when the loss function weights are calculated for each batch. The plot on the bottom compares three different loss function weighting techniques when the batch size is 2 and 32.

4.2. Comparisons

There are few public domain datasets in the field of remote sensing, and in particular, multi-class segmentation public datasets, suitable for training a deep model, are rare. In this section, we show the results of the comparison between our method and the results associated with the only public dataset available [20]. Furthermore, we did an analysis on the use of different ResNet used as backbone of the bottom-up block of the proposed model. The results are reported in Table 3 and show that as the complexity of the bottom-up block increases, the accuracy of the classification improves. We have not used more complex models due to our limited hardware availability, but we should probably get better results using more powerful models.

Table 3. Results on Class Activation Intervals (CAIs) using MSELoss+CELoss while changing the bottom-up backbone ConvNet. All the models were trained on the Munich dataset for 300 epochs.

Backbone	**acc.**	**MSE Loss**
ResNet101	93.48%	no
ResNet101	93.71%	yes
ResNet50	93.62%	yes
ResNet34	92.19%	yes
ResNet18	91.57%	yes
ResNet10	91.37%	yes

In Table 1, we report the comparison results made with what was published in paper of Rußwurm and Korner [20] and we can conclude that our proposed model behaves better from the point of view of all the metrics used in this comparison. Although the cardinality per class is not the same, we can see that the behavior of the (3+2)D FPN is very similar on both the validation and the test set.

4.3. Experiments on CAI and Ablation Study

To evaluate the quality of the Class Activation Intervals predicted by the network, we do not have any ground-truth values and therefore we asked for an opinion from experts. The result produced is in agreement with experts knowledge. For example, from Figure 3 we can see how, for the CAI of the maize class the model has predicted the time interval that goes from May to the beginning of November, while for the *winter wheat* class the model has predicted a CAI that goes from late April until early July despite the NDVI values are still high. This high NDVI value is typically due to weed presence, re-growth after harvesting and a subsequent crop. Indeed, the model identified in these noisy and complex data the time domain that is uniquely related to wheat growth.

To understand how the CAI changes with the variation of the patch on which it is calculated, we have taken into consideration the *winter wheat* class and calculated all the CAI vectors on the entire test set. We show in Figure 8 an aggregate mean value for each patch, and we can see how the time interval associated with this class does not change according to the patch, even if the average activation values $n^c_{i,j,t}$ change slightly. The network used in this experiment uses Resnet 101 as a bottom-up block and the numerical results are reported in Table 1.

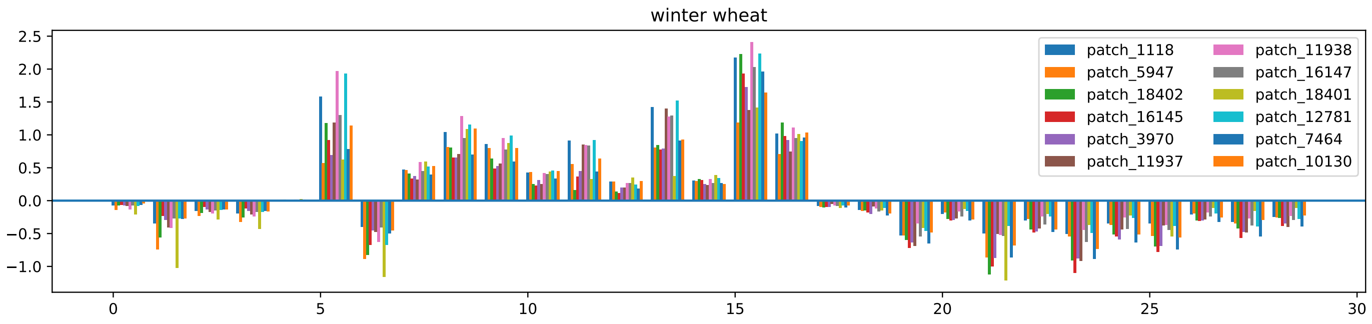

Figure 8. Mean CAIs calculated for all patches of the Munich test set that contain at least 100 pixels labeled with the winter wheat class. The CAI values for each patch were aggregated using the mean. The X axis reports the integer values from 1 up to the number of images used in the time series and therefore is a sampling of the time period analyzed on each series (one year in this paper). The Y axis represents the activation of the output predicted class (CAI) at the date it saw in the input time series.

To understand the effect of learning the NDVI indices with the aim of predicting the CAI, we also conducted an ablation study in which we eliminate the MSE loss during training and compare this result with the same model that uses the loss function. From the numerical results reported in Table 3 we can see that the MSE Loss does not greatly affect the classification performance.

5. Conclusions

The proposed model represents an efficient way to produce crop maps by exploiting spatio-temporal information of S2 data. The proposed (3+2)D FPN performs very well when applied to Sentinel-2 data, over-performing the proposed state of the art solution. The work carried out opens the door to new studies in the field of the comprehensibility of deep learning algorithms in agricultural and environmental applications. We consider that the method described represents a step forward to understand the behavior of deep learning models in agricultural applications. The provision of CAI values at pixel level is a way to assess the robustness of network interpretation of the considered semantic class. If the temporal period (i.e., subset of image time series), considered as importance from the network, agree with known agro-practices (i.e., sowing period of cultivated crops) we can be confident of exportability of the method in other context. Furthermore, the proposed technique for interpreting the activation interval of a class in the time domain is very innovative and can be adapted to other domains that make use of fully convolutive deep models. Moreover, beside the innovative contribution in pattern recognition domain we consider that this encouraging results can be of extreme help for final user. In particular automatic detection of "where and when" crop are cultivated is a fundamental support territorial planner and policy makers. In particular in the European context, Common Agricultural Policy (CAP) is devoted to support farmers by providing subsidies according to cultivated crop and accomplishment of proper agro-practices. Paying Agencies, that have to decide about the farmers' declaration compliance, need automatic external information to support their checking activities (see the Sen4Cap program, http://esa-sen4cap.org/, accessed on 11 July 2021). The proposed segmentation model provides very interesting spatial (where is the crop) and temporal information (when it is cultivated) as a support for user interested in monitoring crop dynamics for a given geographical area. To further contribute to crop monitoring in view of sustainable agriculture and climate friendly practices, a potential next step of the study will be to test the possibility of providing indication on land status (what and for how long) before and after crop cultivation to identify soil management such as presence of cover crops and or residues that are indicator of agricultural conservation practices. In this way the model should also become multi-labeled, as well as multi-classed, for each pixel of the output segmentation map.

Author Contributions: Conceptualization, Ignazio Gallo and Mirco Boschetti; methodology, Ignazio Gallo; software, Ignazio Gallo, Nicola Landro and Riccardo La Grassa; validation, formal analysis, investigation, writing, Ignazio Gallo, Nicola Landro, Riccardo La Grassa and Mirco Boschetti; data curation, Mirco Boschetti. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sochor, J.; Herout, A.; Havel, J. Boxcars: 3d boxes as cnn input for improved fine-grained vehicle recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3006–3015.
2. Liu, J.; Cao, L.; Akin, O.; Tian, Y. Accurate and Robust Pulmonary Nodule Detection by 3D Feature Pyramid Network with Self-supervised Feature Learning. *arXiv* **2019**, arXiv:1907.11704.
3. Simonyan, K.; Zisserman, A. Two-stream convolutional networks for action recognition in videos. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 568–576.
4. Feichtenhofer, C.; Pinz, A.; Zisserman, A. Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1933–1941.
5. Burceanu, E.; Leordeanu, M. A 3d convolutional approach to spectral object segmentation in space and time. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI, Vienna, Austria, 23–29 July 2020; pp. 495–501.
6. Hara, K.; Kataoka, H.; Satoh, Y. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6546–6555.
7. Qiu, Z.; Yao, T.; Mei, T. Learning spatio-temporal representation with pseudo-3d residual networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5533–5541.
8. Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; Paluri, M. A closer look at spatiotemporal convolutions for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6450–6459.
9. Sentinel Dataflow from Copernicus Program. 2021. Available online: https://www.copernicus.eu/en (accessed on 11 July 2021) .
10. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.

11. Seferbekov, S.S.; Iglovikov, V.; Buslaev, A.; Shvets, A. Feature Pyramid Network for Multi-Class Land Segmentation; In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 2018; pp. 272–275.
12. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
13. Isensee, F.; Jäger, P.F.; Kohl, S.A.; Petersen, J.; Maier-Hein, K.H. Automated design of deep learning methods for biomedical image segmentation. *arXiv* **2019**, arXiv:1904.08128.
14. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning deep features for discriminative localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2921–2929.
15. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Object detectors emerge in deep scene cnns. *arXiv* **2014**, arXiv:1412.6856.
16. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
17. Kirillov, A.; Girshick, R.; He, K.; Dollár, P. Panoptic feature pyramid networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 6399–6408.
18. Zhu, L.; Deng, Z.; Hu, X.; Fu, C.W.; Xu, X.; Qin, J.; Heng, P.A. Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 121–136.
19. Rousel, J.; Haas, R.; Schell, J.; Deering, D. Monitoring vegetation systems in the great plains with ERTS. In *Proceedings of the Third Earth Resources Technology Satellite—1 Symposium*; NASA: Washington, DC, USA, 1974; pp. 309–317.
20. Rußwurm, M.; Körner, M. Multi-temporal land cover classification with sequential recurrent encoders. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 129. [CrossRef]
21. Rußwurm, M.K.M. Munich Dataset. 2018. Available online: https://github.com/tum-lmf/mtlcc-pytorch (accessed on 11 January 2021).
22. McHugh, M.L. Interrater reliability: The kappa statistic. *Biochem. Medica Biochem. Medica* **2012**, *22*, 276–282. [CrossRef]
23. Robbins, H.; Monro, S. A stochastic approximation method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
24. Gallo, I.; La Grassa, R.; Landro N.; Boschetti M. Pytorch Source Code for the Model Proposed in This Paper. 2021. Available online: https://gitlab.com/ignazio.gallo/sentinel-2-time-series-with-3d-fpn-and-time-domain-cai (accessed on 11 July 2021).
25. Cui, Y.; Jia, M.; Lin, T.Y.; Song, Y.; Belongie, S. Class-Balanced Loss Based on Effective Number of Samples. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.