Deep learning in the built environment: automatic detection of rooftop solar panels using Convolutional Neural Networks

Deep learning in the built environment: automatic detection of rooftop solar panels using Convolutional Neural Networks

Roberto Castello[1], Simon Roquette[1], Martin Esguerra[1], Adrian Guerra[1] and Jean-Louis Scartezzini[1]

[1]Solar Energy and Building Physics Laboratory, Ecole Polytechnique Fédérale de Lausanne, CH-1015, Lausanne, Switzerland

roberto.castello@epfl.ch

Abstract. Mapping the location and size of solar installations in urban areas can be a valuable input for policymakers and for investing in distributed energy infrastructures. Machine Learning techniques, combined with satellite and aerial imagery, allow to overcome the limitations of surveys and sparse databases in providing this mapping at large scale. In this paper we apply a supervised method based on convolutional neural networks to delineate rooftop solar panels and to detect their sizes by means of pixel-wise image segmentation. As input to the algorithm, we rely on high resolution aerial photos provided by the Swiss Federal Office of Topography. We explore different data augmentation and we vary network parameters in order to maximize model performance. Preliminary results show that we are able to automatically detect in test images the area of a set of solar panels at pixel level with an accuracy of about 0.94 and an Intersection over Union index of up to 0.64. The scalability of the trained model allows to predict the existing solar panels deployment at the Swiss national scale. The correlation with local environmental and socio-economic variables would allow to extract predictive models to foster future adoption of solar technology in urban areas.

1. Introduction

Switzerland has ambitious goals for increasing its use of renewable energy and reducing carbon footprints. In order to ensure that future urban developments are in line with global objectives, there is a constant need to monitor the transformation of the built environment and to improve the representation of urban areas in power grid models. The rapid deployment [1] of decentralized power generation in urban areas, through solar photovoltaic (PV) technology on building rooftops, calls for a comprehensive database with locations and sizes of existing installations in urban areas. Moreover, the need for estimation of solar rooftop PV potential at the Swiss national scale [2][3], which is a critical input for utility planning and energy market policies, requires a precise assessment of the available rooftop surface. This assessment will allow to identify and exclude roof superstructures from the calculation, for example. Furthermore, such a comprehensive appraisal can become extremely helpful when designing distributed energy systems, such as energy hubs [4].

To the best of our knowledge, few countries today have updated databases of their solar PV installations. The Open PV project [5] is an example which provides, through a collaborative effort from public, government and industry, a comprehensive accessible online database. Voluntary surveys, self-reports [6] are also a possibility, but often they are incomplete or they can become quickly obsolete at

the moment in which they are made accessible. Machine Learning (ML) approaches based on Convolutional Neural Networks (CNNs) [7] combined with satellite and aerial imagery can be used to overcome these limitations and to establish a benchmark for automated classification of a plethora of shapes in urban areas. Some work has been already done in this direction (e.g. [8], [9] and [10]) but, except for the United States, no country has yet come up with a comprehensive national database.

Computer Vision (CV) techniques have found great usage in diverse areas such autonomous driving [11], facial recognition [12] and medical imaging [13]. In this paper, we investigate the potential of CV methods and ortho-rectified images in predicting pixel-wise *PV* class versus *no-PV* class in high resolution images. Potentially, this would allow to extract the surface of installed rooftop solar PV for any building of Switzerland, provided enough computational resources and aerial imagery coverage.

2. Convolutional Neural Networks for object segmentation

In computer vision, CNNs are designed to process data that come in the shape of multi-dimensional arrays, such as an RGB image. Such images are usually structured in three (Red, Green and Blue) two-dimensional arrays, called *channels*, which contain pixel values encoded as 8-bit integer. Each pixel can thus assume an integer value between 0 and 255, giving origin to the pixel colours.

Convolution is an operation where a *kernel* (matrix of scalars) is applied to an array of pixels, in order to generate another array. The resulting elements of the new array are linear combinations of the kernel and of the original one. A convolutional layer in a CNN tunes the scalar values of the kernel in order to minimize a certain cost (or *loss*) function. Each neuron in the layer has a weight, represented by a *filter* which works to extract a *feature map* of a channel. The sequence of visible (input and output) and hidden layers define the main network architecture. Another key block of a CNN is represented by the *pooling* layers. The role of the pooling layer is to merge semantically similar features into a single one [7] in order to reduce the computational load and the number of parameters, thus reducing the risk of overfitting. The pooling layer has no weight and it only aggregates the inputs using a function like the max of each kernel's input values before shipping them to the next layer.

The most popular CNN architecture used for fast and precise segmentation of images is *U-Net* [14]. It can be seen as an encoder-decoder consisting of two phases. The first one is a down-sampling phase that combines convolution with max-pooling layers in order to capture features from the smallest scale to the image scale. The second phase consists of up-sampling the result of the first one in order to put the features back into the format of the original image. The output of each stage of the down-sampling phase is fed directly to the corresponding up-sampling phase to avoid separate training of encoder and decoder. Figure 1 shows a graphical representation of the network and of a typical max-pool layer operation.

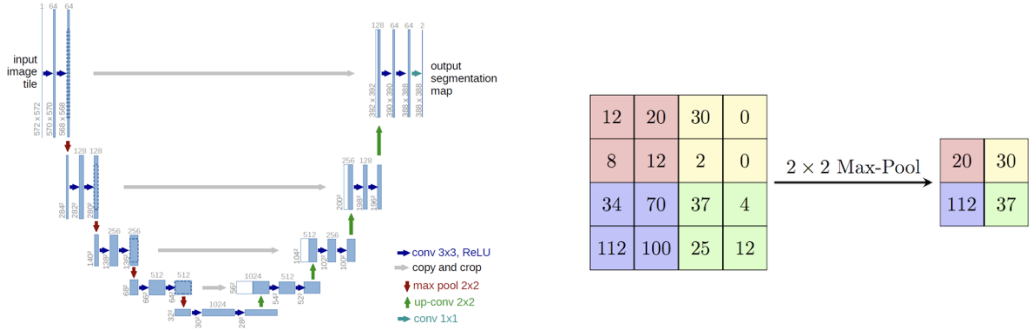

Figure 1: *(Left) Example of U-Net architecture [14]. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of each box. The image pixel size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. (Right) Example of 2x2 Max-Pool operation. Each max is taken over the 4 numbers in the kernel.*

3. Network training

3.1. Input dataset

We use ortho-rectified 8-bit RGB images of Switzerland provided by the Swiss Federal Office of Topography in TIF format, collected during the year 2013. They come in tiles of 17500x12000 pixels, with a spatial resolution of 0.25x0.25 m^2. The large size of the images and the relatively small pixel occupancy in a tile of rooftop solar PVs (around 3%) make the detection very challenging. Therefore, each image tile is further divided into several 250x250 pixels sub-images in PNG format. They are chosen to sample different urban and rural settings (mostly in the areas of Geneva, Lausanne, Thun and Bern), in order to have the most inclusive variety of rooftop PV shapes and types. Being a supervised learning task, we manually classify and segment PV panels over about 700 images by means of a Python-based tool using OpenCV library [15] in order to create the labelled sample. We also add a small percentage of images which do not contain any PV panel to the training and labelled set. We do this operation to reinforce the rejection of false positives, expanding the final dataset to 780 images. Finally, we artificially enlarge the dataset using label-preserving transformations [16]. We create two lighting sets and three ninety-degrees rotation sets for each image. By doing so, we potentially augment the original training set by a factor six (780x2x3 = 4680). We train our model using only the original set and then by adding the augmented ones, at first separately for rotations and lighting and afterwards by using them together.

3.2. Architecture

In order to select the most performing U-Net configuration, we tune the hundreds of thousands of model's parameters using a HPC cluster equipped with 16 GPU accelerated nodes, each with 4 K40 NVIDIA cards. We start from the three-channels input images and we first perform a contracting path, followed by an expansive one. The first convolutional layer is made of 32 filters of dimension 256x256 (this size is obtained by using the *zero-padding* technique [18] on the original 250x250 image). The kernel size used is 3x3. In a second stage a 2x2 max-pooling layer reduces the size to 128x128x32 and a convolutional one expands the layers to 128x128x64. The same sequence is applied for the third contracting stage, bringing the lowest layer of the U-net to a size of 64x64x128. At this point, the expansive path starts in a specular way, where padding and pooling operators are replaced by cropping and up-sampling operators respectively. In the expansive phase, each stage receives the feature channels of the corresponding contracting layer and merging it together allows the network to propagate the image context information to higher resolution layers. In total there are 471586 trainable parameters.

3.3. Training

Choosing a good loss function is crucial in image segmentation task. A pixel-wise categorical cross-entropy [19] results in a network that always predicts no-PV pixels, as our classes are very unbalanced. We therefore consider a *weighted* pixel-wise categorical cross entropy function, by assigning larger weight to the false negative loss (predicting PV instead of no-PV). Finding optimal weights for the loss is challenging, due to the trade-off between false positive and false negative. We tune the loss function parameters by sampling different values for the weight pairs from (1, 1.5) up to (1, 9) with step of 1.5 (being the first one for false positive and the second one for false negative). The pair giving the best convergence is (1, 5), which we utilize in the loss function.

We do not start from a pre-trained model; hence we train the algorithm starting from a random set of weights, using batches of 32 images and 80% of the input images. The remaining 20% is used for validation. Adding to the training set the rotations and lighting sets described in Section 3.1 clearly helps the model convergence, enhancing to 4680 the number of images finally used. We test *Adam* and *Stochastic Gradient Descent* optimizers [20], seeing better convergence with Adam, with a learning rate of 0.1. We find that 75 epochs are the optimal choice for preventing model overfitting as, by increasing to 100 and 125, the learning curves for training and validation loss start to diverge. We add dropout layers [18] with a rate of 0.2 after each of the layers described in Section 3.2 in order to further stabilize the model output. Figure 2 (left) reports the evolution of the model loss as a function of the number of epochs used in the training phase, showing an overall good level of convergence and no overfitting. Although not affecting the global convergence, a few spikes are still visible in the training and validation

loss curves. They may be removed or further reduced by using a larger dataset or by increasing the images batch size used during the training phase.

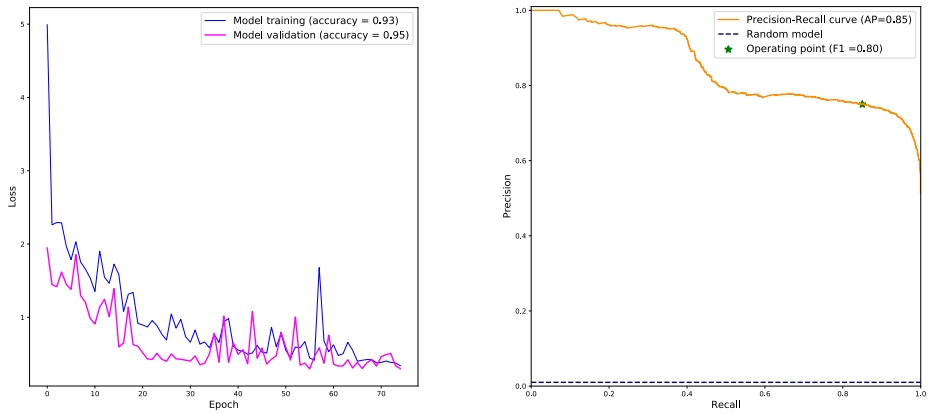

Figure 2: *(Left) Performance of the trained model: loss as a function of the epochs on the training (blue) and validation (magenta) datasets. (Right) Model skill in the form of Precision-Recall curve (orange) compared to a random model (dashed blue line), together with the Average Precision score (AP) and the chosen operating point (giving a F1 score of 0.8)*

4. Results and discussion

We apply the trained network to a few test images randomly selected in a non-overlapping area with the one covered by training set and we quote the performance score. A widely used metric in classification problems is the *accuracy*, i.e. the percentage of pixels in the image which are correctly classified, both as true positive or true negative. The accuracy metric can sometimes provide misleading results when the pixel class (PV) presence is small within the image, as the measure will be biased in mainly reporting how well the true negative cases (no-PV) are identified. A more suitable metric for image segmentation is the *Intersection over Union* (IoU), which consists of computing the common pixel area between the prediction and the ground-truth and dividing it by the area of union. Therefore, in tasks like ours where the pixel percent of the target class in the image is a tiny fraction, IoU rather than accuracy is a more robust estimator of the model performance. Figure 2 (right) also measures our final model skill in terms of Precision-Recall (PR) curve, more suitable than ROC curves for tasks exhibiting a large class imbalance, as in the case of our binary classification problem (PV vs no-PV pixels). Based on PR curve, we chose the probability threshold of the prediction array to be 95%, resulting in a F1 score (harmonic mean of precision and recall) of 0.8. This gives an IoU score of 0.64 and an accuracy of 0.94.

Figure 3 provides a visual assessment of the algorithm performance in a rural area, randomly chosen in the municipality of Lausanne. Noticeably, the model can identify with rather well-localized boundaries the solar PV in the image tile although some false positives are still present, in particular ground objects which happen to be similar to PV panels in terms of both patterns and their surroundings. Incorporating more images which do not contain any PV panel should further reduce such error.

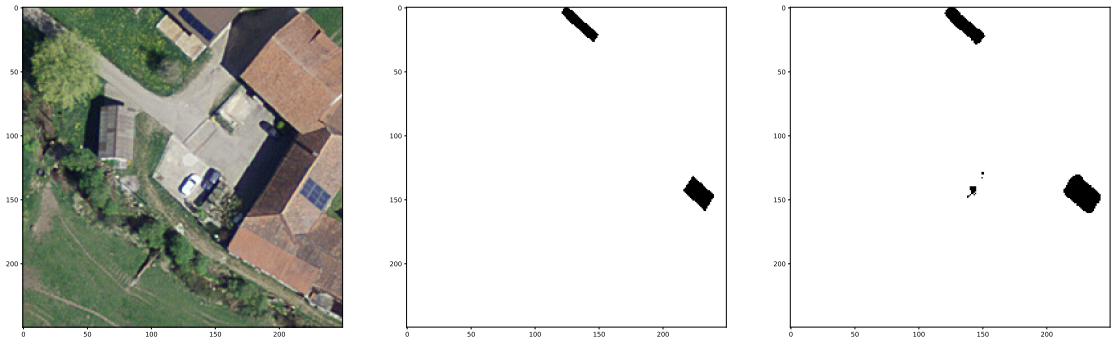

Figure 3: *Left: Real image (size: 250x250 pixels) where solar PV installations are present (randomly extracted from the test dataset). Centre: Labelled image showing the ground truth (black pixels = PV, white pixels = no-PV). Right: Results of the predictive model*

Further, we notice that an increase of the relative PV pixels fraction in the training images might certainly be beneficial in order to further improve the model and to augment its detection precision. This can be obtained by dividing the images in smaller patches and training only by using those where at least one PV is present. Equivalently, also up-sampling the original image through interpolation to achieve e.g. half of the original resolution should have a similar effect. Another promising approach would be to turn the binary classification (PV versus no-PV) into a multi-class pixel-wise classification task. This technique has been successfully adopted in [9] and it consists of having, for each pixel, a class indicating the distance to the closest PV. Negative values represent the distance outside the PV borders (belonging to class '0') and positive values the distance inside the PV borders. Being a segmentation task, the network might also benefit of an additional layer, called *Sobel filter* [17], which detects image edges by computing the horizontal and vertical derivatives of pixels' color intensities. This would help the model to learn faster, as it provides information on the edges contained into an image and it would also enhance its segmentation precision.

In addition to the delineation of solar PV installations, the framework here developed can serve as basis for the automatic detection of a wide range of urban objects (building types, green areas, rooftop superstructures, etc..) which could significantly increase the mapping capability at the Swiss national scale. Also, the outcome of such mapping (in our case the installed solar PV rooftop area at the neighbourhood scale) can be further correlated with local socioeconomic factors like population, income and level of education, as already attempted in [10]. Using them as target and features we can build predictive models which will help to track the evolution of future solar technology in Swiss urban and rural areas. Also, whenever a new set of aerial images for a certain year is released, it will be possible to automatically track the evolution and to populate national and local databases.

5. Conclusions

This paper reports on the first ever attempt in Switzerland of mapping locations and sizes of existing solar rooftop installations by applying machine learning techniques to aerial images. A convolutional neural network in the form of a U-Net has been proposed and its training and optimization has been realized with the objective of improving the model performance. The trained model applied to a set of test images shows an accuracy of 0.94, together with an Intersection over Union score of up to 0.64. Results show the promise and limits of our approach, while leaving room for further improvements. This framework can be extended to detect other shapes in the built environment. Ultimately, the results can be used as input dataset for training regression models that are able to capture the relationship between socioeconomic features and PV deployment, with the aim of fostering solar technology adoption in current and future urban and rural areas.

Acknowledgments
Authors would like to thank Nicola Varini from EPFL IT division (SCITAS) for the technical support during the code optimization phase. The access to the HPC facilities is granted by the CADMOS collaborative project. This research project is financially supported by the Swiss Innovation Agency Innosuisse under the Swiss Competence Centre for Energy Research SCCER FEEB&D and by the *HyEnergy* project under the NRP 75 "Big Data" series of the Swiss National Science Foundation.

References

[1] Swiss Federal Office of Energy. Electricity production and consumption in 2017 n.d.
[2] Assouline D, Mohajeri N, Scartezzini J-L. Large-scale rooftop solar photovoltaic technical potential estimation using Random Forests. Applied Energy 2018;217:189–211. doi:10.1016/j.apenergy.2018.02.118.
[3] Walch A, Castello R, Mohajeri N, Guignard F, Kanevski M, Scartezzini J-L. Spatio-temporal modelling and uncertainty estimation of hourly global solar irradiance using Extreme Learning Machines. Energy Procedia 2019;158:6378–83. doi:10.1016/j.egypro.2019.01.219.
[4] Mavromatidis G, Orehounig K, Carmeliet J. Evaluation of photovoltaic integration potential in a village. Solar Energy 2015;121:152–68. doi:10.1016/j.solener.2015.03.044
[5] The Open PV Project n.d. https://openpv.nrel.gov/
[6] PVOutput n.d. https://www.pvoutput.org/
[7] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521:436–44. doi:10.1038/nature14539.
[8] Huang Z, Mendis T, Xu S. Urban solar utilization potential mapping via deep learning technology: A case study of Wuhan, China. Applied Energy 2019;250:283–91. doi:10.1016/j.apenergy.2019.04.113.
[9] Yuan J, Yang HL, Omitaomu OA, Bhaduri BL. Large-scale solar panel mapping from aerial images using deep convolutional networks. 2016 IEEE International Conference on Big Data (Big Data), 2016, p. 2703–8. doi:10.1109/BigData.2016.7840915.
[10] Yu J, Wang Z, Majumdar A, Rajagopal R. DeepSolar: A Machine Learning Framework to Efficiently Construct a Solar Deployment Database in the United States. Joule 2018;2:2605–17. doi:10.1016/j.joule.2018.11.021.
[11] Real-time object detection for "smart" vehicles - IEEE Conference Publication n.d. https://ieeexplore.ieee.org/document/791202
[12] DeepFace: Closing the Gap to Human-Level Performance in Face Verification n.d. https://www.computer.org/csdl/proceedings-article/cvpr/2014/5118b701/12OmNzFdt9h
[13] Current Methods in Medical Image Segmentation | Annual Review of Biomedical Engineering n.d. https://www.annualreviews.org/doi/full/10.1146/annurev.bioeng.2.1.315
[14] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. ArXiv:150504597 [Cs] 2015.
[15] Open Source Computer Vision Library. Contribute to opencv/opencv development by creating an account on GitHub. OpenCV; 2019.
[16] Best practices for convolutional neural networks applied to visual document analysis - IEEE Conference Publication n.d. https://ieeexplore.ieee.org/document/1227801
[17] Vincent O, Folorunso O. A Descriptive Algorithm for Sobel Image Edge Detection, 2009. doi:10.28945/3351.
[18] Convolutional Layers - Keras Documentation n.d. https://keras.io/layers/convolutional/
[19] Losses - Keras Documentation n.d. https://keras.io/losses/
[20] Optimizers - Keras Documentation n.d. https://keras.io/optimizers/