Detection of snow-coverage on PV-modules with images based on CNN-techniques

Dennis Hepp[1], Sebastian Hempelmann[2], Grit Behrens[3] and Werner Friedrich[4]

Abstract: The transition from fossil fuels to renewable energy is considered as very meaningful to mitigate climate change. To integrate weather-dependent energies firmly into the power grid, a forecast of the energy yield is very important. This paper is about renewable energy generation by photovoltaic (PV) systems. The yield of PV-systems depends not only on weather conditions, but in wintertime also on the additional factor "snow cover". The aim of this work is to detect snow cover on photovoltaic plants to support the energy yield forecast. For this purpose, images of a PV-plant with and without snow cover are used for feature extraction and then analyzed by using a convolutional neural network (CNN).

Keywords: convolutional neural network; machine learning; python; image recognition; snow detection; photovoltaic

Addresses Sustainable Development Goal 7: Affordable and clean energy

1. Introduction

The climate change affects us all. If it is not sufficiently curbed, our planet could become uninhabitable in the future. [IP22] To prevent this, for example, CO_2 emissions must be drastically reduced. [IP21] With energy consumption on the rise, it only makes sense to reduce CO_2 emissions in the energy production sector as well. This can be achieved by switching from fossil fuels to renewable energies. [IE22] One of the weather-dependent renewable energy sources is solar power, which is generated by PV-systems. These plants can be monitored and based on this data it is also possible to make forecasts for the future. [He20] For the forecast, some factors must be considered. PV-systems depending on weather conditions and location. Plenty of sunshine is an indicator of high electricity yields. However, these can be reduced by objects that cast shadows such as trees or clouds, soiling on the plant, or age-related side effects. [Qu11] To make an accurate energy yield forecast, these factors, as well as the prevailing weather, must be included in the calculation. If the PV-system does not supply as much electricity as predicted, an error message is generated in the monitoring system. Which either will be used to coordinate

[1] Bielefeld University of Applied Sciences, Faculty of Engineering and Mathematics, Langer Weg 9a, 33332 Gütersloh, Germany, dennis.hepp@fh-bielefeld.de

[2] Bielefeld University of Applied Sciences, Faculty of Minden Campus, Artilleriestrasse 9, 32427 Minden, Germany, sebastian.hempelmann@fh-bielefeld.de

[3] Bielefeld University of Applied Sciences, Faculty of Minden Campus, Artilleriestrasse 9, 32427 Minden, Germany, grit.behrens@fh-bielefeld.de

[4] DSL & Photovoltaikberatung, Frankfurter Str. 9b, 65551 Limburg, Germany, wf@etechnikfreak.de

maintenance work or improving the forecast. This is the aim of the research project “PV digital 4.0”, which is funded under grant number 13FH020PX6. This work deals with the snow-coverage on PV plants and is structured as follows, in chapter 2 the aim of the work will be described. Further, this paper involves the detection of snow coating on PV-systems based on images and a convolutional neural network (CNN), which is discussed in detail in Chapter 3. The results are discussed in Chapter 4 and Chapter 5 concludes and gives an outlook for the future.

2. Aim of the work

The goal of this work is to detect snow coverage on PV systems using pictures. To put it into a larger context, this detection is used to label the monitoring data. The labelled data allows us to train a standalone neural network, which can detect snow on plants independent of their location and orientation towards the sun, just using the monitoring data together with weather data. [HBB22]

This is one of the in chapter 1 mentioned factors, which need to be considered in the project “PV Digital 4.0” to increase the accuracy of the yield prediction.

3. Materials and methods

In this work a software based on *Python* has been developed using modules like *datetime* and *glob*. For downloading image data *selenium 4.1.5* was used, to pre-process the data the Python library *OpenCV 4.5.5.64* was utilized and to train a network which should finally detect snow on the PV-systems a CNN was built with *keras 2.9.0* (based on *TensorFlow 2*).

3.1 Overview

A convolutional neural network (CNN) is a special type of neural networks (NN). The neural network is the model used in deep learning. It consists of several layers in which a selected number of neurons are located. These neurons have trainable weights and are connected to each other. How these connections look like is determined by the type of neural network. The architecture, i.e. the number of layers and neurons, is developed by the programmer. Typical neural networks learn a global pattern. CNNs, on the other hand only ever learn a local pattern, so it can recognize edges, textures, etc. in images and is perfectly designed for image recognition. [Ch17] To make this possible, CNN comes up with some special methods, which are explained in more detail in following subsections.

3.2 Convolutional neural network

Convolutional neural networks are typically used for image recognition. The CNN takes an image as their input, and so-called kernels (or filters) are applied to extract features. The input image data for the CNN is handed over with the shape (image_height, image_width, image_channels). A grayscale picture has just one channel, which means every pixel in the picture can be described with one value. These values are the input for the CNN. The colored pictures used in this work come with three channels (RGB: red, green, blue). The kernel is applied on every channel and then summed up. Kernels are matrices and the parameters are trained like weights in a neural network. [Ch17] This 3-D grid is typically processed in the "convolutional block", which is a combination of the mathematical operation "convolution" (convolutional layer) and the "pooling" method. This will be repeated several times. [GBC16] Throughout the convolutional process, the first convolutional layer can learn small patterns, like edges. The next convolutional layer will combine these small patterns to learn bigger, combined patterns. [Ch17] Once a pattern was learned, the CNN will be able to recognize this pattern in any part of the image (equivariant representations). After passing through the convolutional blocks, the output data will be modified with "flattening", so the output can be classified by an MLP. [Fr19] The mentioned methods are described further in the following paragraphs.

Convolution

The convolution is a mathematical operation, which takes the input and overlays a so-called kernel. This kernel is a matrix, which can also be seen as weights from a typical NN and will be used for the whole CNN (parameter sharing). It is smaller than our input, which leads to "sparse interaction" and will be used for each channel. So, the kernel is put on the input. The single values are multiplied with each other and then summed up. Once completely shifted over the whole input a new, smaller matrix, the "feature-map" results. If the down scaling is not wanted, (zero-) "padding" can be used, which won't be discussed further in this work. [Bi08]

Pooling

A part of the convolutional block is pooling. The aim of a CNN is to learn bigger patterns, just using the convolution operation would take a long time to recognize such big patterns. Therefore, the data is compressed and generalized in the pooling layer to speed up the process. The most used pooling approach is the "max-pooling". Max-pooling is modifying the output further by looking at a certain window in the feature-map and only takes the biggest value in that to create the output of the convolutional layer, our final feature-map. [Fr19]

Flattening

Flattening is used to reorder the entries of the 3-D grid resulting from the convolutional layers. The grid will be stringed together to a vector, which then can be used to for the classification task with the MLP. [Fr19]

3.3 Data base

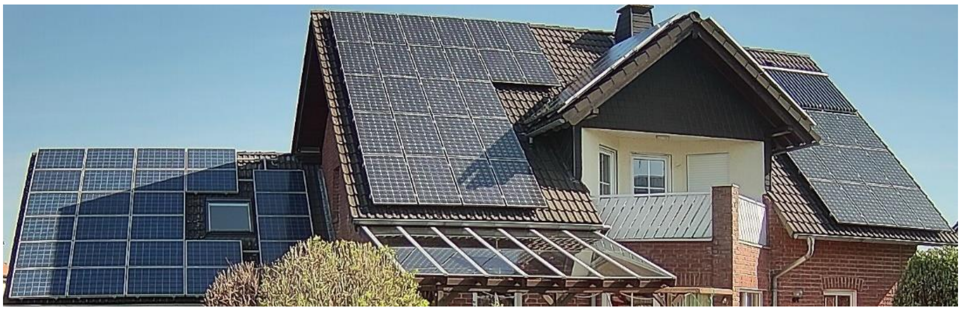


Fig. 1: Image taken by the camera (due privacy, this picture was cropped).

For the described experiment, a camera was set up in the center of Germany 173m above sea level in Hesse (coordinates: 50.37194660775961, 8.122048403586822). The camera is pointing to a private roof with a PV-plant on it (shown in Fig. 1) and is accessible via Wi-Fi. A script was written, which can connect to the interface of the camera and allow us to receive images. The images were sent as HTTP-request to a LINUX server running the script. Via cronjob, this process was executed every five minutes during the day from the end of November 2021 to April 2022. Due to transmission errors, the camera was not always available, so there were gaps in the data. In total 19.730 images could be collected in the experiment. Among these, 111 of the images show snow cover on the PV-system. Afterwards, the visible parts of the plant were cut out in rectangles (see Fig. 2) with OpenCV and were saved. This resulted in a total of 256 usable images of a snow-covered PV-system and a lot more non-snow-covered images, that could be used for training.

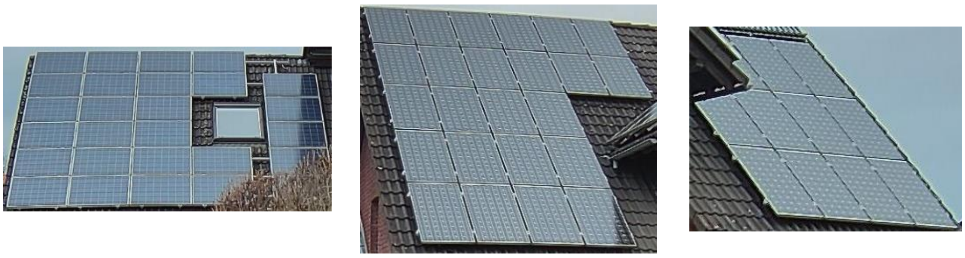
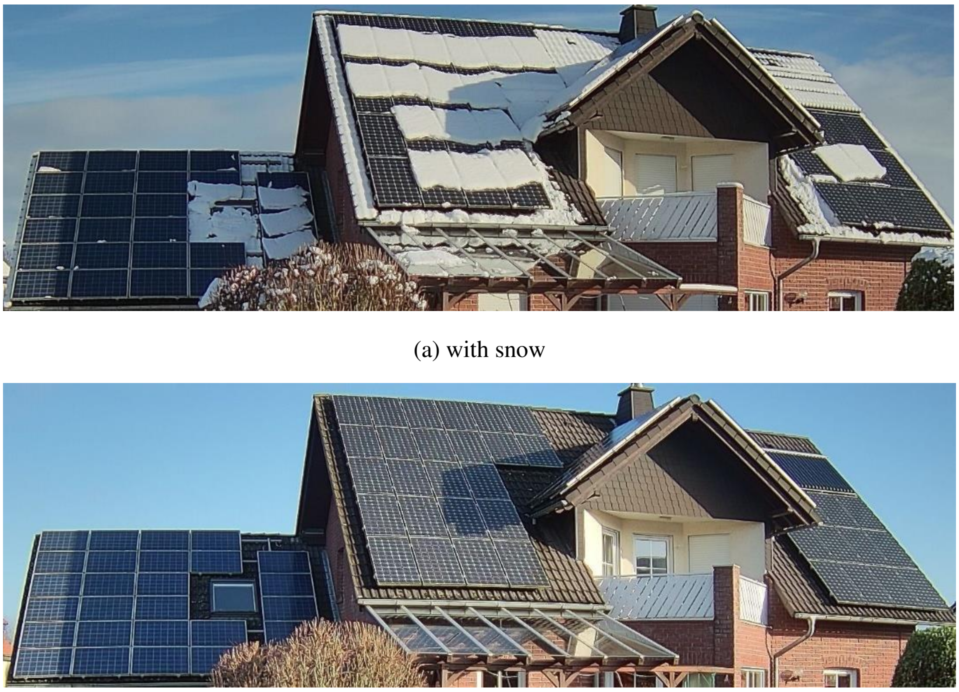


Fig.2: The PV-plant can be divided into three smaller parts.

3.4 Data selection

For good results, the training data set should be well-balanced with the same number of pictures with snow cover and without snow cover. For this purpose, care must be taken in selecting days with similar weather conditions, since the position of the sun affects the light conditions in the image, as well as the reflection on the solar panels. In addition, depending on the time of day, the roof, and dormer casts a shadow (e.g. Fig. 3) on some PV-modules.



(a) with snow

(b) without snow

Fig. 3: In between these two sunny-day pictures are three days. This grants us similar conditions, like the angle of the shadow.

3.5 Pre-processing

Unfortunately, in the period in which the pictures were taken, it didn't snow often, so the amount of data is small. Good results come naturally from good data, so the mass of qualitative images is an important factor. To increase the number of images indirectly, the *ImageDataGenerator* of *keras* was used, which randomly rotates, shifts, shears, and zooms into the training images during the training process of the model and changes the size of the images to 512 x 512. This is also called “data augmentation”. The data augmentation is applied for each epoch. So, in each epoch, our CNN gets a randomly generated dataset based on the rules set in the *ImageDataGenerator* and our base dataset. [Fr19]

3.6 Data analysis

For image recognition we used a CNN (shown in Fig. 4) as described in 3.2. The CNN takes an input with the shape (512, 512, 3) because we rescaled all images in 3.5. We used RGB pictures, therefore we needed to set the image_channel to three. In total, we used four convolutional blocks, a 2-D convolution with the activation function "ReLU" followed by a 2-D MaxPooling. The Kernel for the convolution was given the size of 3 x 3 and the MaxPooling window is set to 2 x 2. This helps us to focus on the important features. The first convolution is set to give us an output dimension of 32, so our first output shape will be (510, 510, 32). After the MaxPooling we have a shape of (255, 255, 32). The second convolution is set to 64 and the third and fourth to 128. At the end we got a shape of (30, 30, 128) this will be flattened to a 1-D vector with the shape of (115200), which is basically the multiplication of the shape of our last convolutional layer (30*30*128 = 115200). Our dense layer of the MLP is set to an input of 512 neurons, also with the "ReLU" activation function. Followed by a dense layer with one neuron and the "sigmoid"-function, which will take the input to give an output between 0 and 1. A threshold is then used for the classification, every output above the threshold will be classified as 1 for "snow-coverage" and every output below the threshold will be classified as 0 for "no snow-coverage". For the loss function, we used the "binary cross-entropy", which is common for CNNs. [Fr19] The common optimizer "Adam" was chosen for the optimization task of this network. The dataset was split into three data sets: 70% train data, 20% validation data and 10% test data.

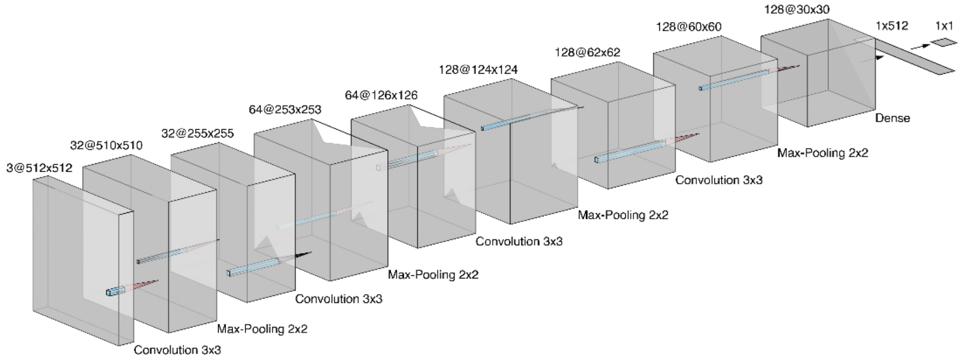


Fig. 4: This is a visualization of the used CNN. The first cube represents our input layer with the shape (512, 512, 3). The convolution and the MaxPooling process is visualized by the blue cutouts pointing to the next cube. Every convolution step is increasing the third dimension, which were the RGB-values of the input image in the first step. The MaxPooling is decreasing the first and second dimension, which is the height and width

of the image. After a few steps e.g., the sixth cube got a shape of (124, 124, 128), which means the image got compressed to a height and width 124 x 124 with 128 features. After the eighth cube, the data will be flattened and commit it to our first Dense layer with 512 neurons.

4. Results

The results of the training are visualized in Fig. 5. The accuracy while training achieved more than 96%. At the end, the CNN was able to predict 100% of the validation data set. The loss value of the training was around 0.16 and for the validation set 0.07. Using the test data to test the CNN leads us to 92% accuracy. The gap between the accuracy of the test and train data set is caused by overfitting. Also, overfitting led our valid prediction to 100%.

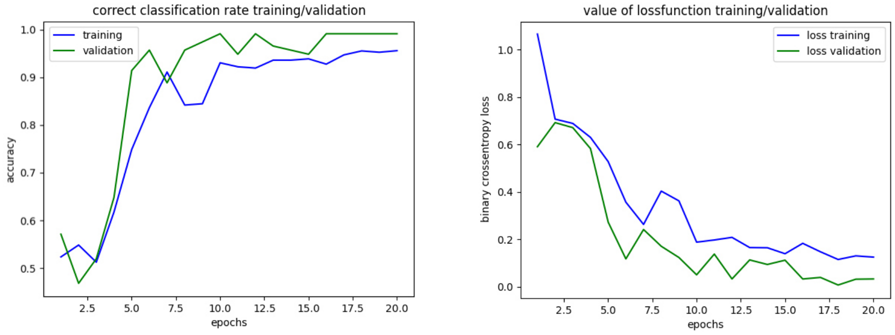


Fig. 5: Accuracy and loss visualized. The accuracy value is in between 0 and 1, which can be transformed into %. This value shows how much percent of the data set was classified correctly. The loss value is given by our loss function (binary cross-entropy), it is the feedback-signal for the CNN, which is used in the training process. The aim of the CNN is to minimize this value while training.

Due to randomly generated test, train, and valid data sets one of the rare cases, having sunny weather with and without snow (shown in Fig. 3), wasn't well distributed. So, the case "with snow" on the PV-plant and "sunshine" made it into the train but not into the valid data set, while "sunshine without snow" on the other hand made it into both sets. Which means the net couldn't validate the special case shown in Fig. 3a. As we used images with the same weather conditions (described in 3.4), the CNN could have learned to recognize sunshine/shadow pictures as no snow-coverage, even when there was snow (see Fig. 6).

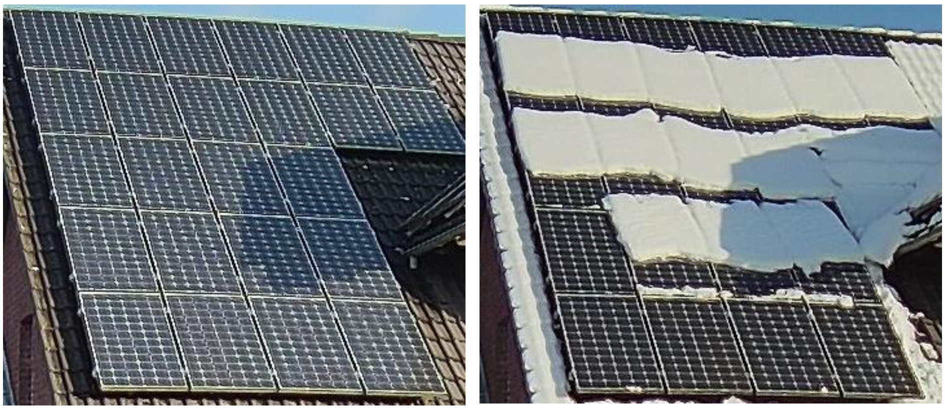


Fig. 63: Both pictures are from the test data set. The left one is showing the class "no snow-coverage", the right one shows "snow-coverage". Both pictures were classified as "no snow-coverage" by our trained model.

5. Conclusion

Even with a small data set we were able to get good results. Overfitting is a problem for our CNN because snow on PV-plants won't ever have the same features as shapes and density. For this, some techniques like the data augmentation mentioned in 3.5 or adding a “Dropout”-layer can be used to prevent this problem. This CNN will be improved with new data from the next winter (in 2022/2023). With the new database, we will increase the accuracy of the CNN. The plan is to take pictures with more cameras on an industrial PV-plant in Germany, which is higher above the sea level and tends to have more snow during wintertime. The next step for the automation of the snow data labeling is either to improve the CNN to automatically detect the snow coverage in percent or to develop an algorithm, which can calculate the snow amount on the plant. With this automated labeling process and with more data, we can also improve the work in [HBB22] and train a better model to recognize snow coverage on PV-plants just using the monitoring and weather data.

Bibliography

[Bi08] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, 2006.

[Ch17] Chollet, F.: Deep Learning with Python. Manning, 2017.

[Fr19] Frochte, J.: Maschinelles Lernen. Hanser, 2019.

[GBC16] Goodfellow, I.; Bengio, Y.; Courville, A.: Deep Learning. MIT Press, 2016.

[IE22] IEA, International Energy Agency, Global electricity demand is growing faster than renewables, driving strong increase in generation from fossil fuels, https://www.iea.org/news/global-electricity-demand-is-growing-fasterthan-renewables-driving-strong-increase-in-generation-from-fossil-fuels, accessed: 12/07/2022.

[He20] Hempelmann, S. et al.: Evaluation of unsupervised anomaly detection approaches on photovoltaic monitoring data. In: 47th IEEE Photovoltaic Specialists Conference (PVSC). Calgary, AB, Canada, 15.06.2020 - 21.08.2020: IEEE, pp. 2671–2674.

[HHB22] Hepp, D.; Hempelmann, S.; Behrens, G.: Detektion von Schneebedeckung mittels stringbasierter Monitoringdaten und Methoden des Maschinellen Lernens. In: 37. PV-Symposium. Conexio-PSE GmbH, Pforzheim, pp. 777-788, 2022.

[IP21] IPCC, 2021: Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge University Press, Cambridge, 2021.

[IP22] IPCC, 2022: Climate Change 2022: Impacts, Adaptation, and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge University Press, Cambridge, 2022.

[Qu11] Quaschning, V.: Regenerative Energiesysteme. Technologie - Berechnung – Simulation. Hanser, 2011.