

Towards more reliable flux density prediction using uncertainty quantification of neural networks

Leon Tim Engelbert Sievers^a, Daniel Maldonado Quinto^a, Bernhard Hoffschmidt^{a,b}

^a DLR, Institute of Solar Research, Linder Höhe, 51147 Köln, Germany

^b RWTH Aachen University, Chair of Solar Technology, Linder Höhe, 51147 Köln, Germany

ARTICLE INFO

Keywords:

Solar power tower
Heliostat
Deep learning
Flux density prediction
Uncertainty quantification

ABSTRACT

Solar tower power plants represent a renewable source of dispatchable energy. By concentrating incoming sunlight via tens of thousands of heliostats, they render large amounts of energy in the form of heat that is usable as a source of electricity or in an industrial context. In this work, we propose an uncertainty-aware prediction framework using Monte-Carlo Dropout and a random forest classifier to flag unreliable flux predictions of said heliostats.

Precise knowledge about the shape and position of the heliostat's reflections is crucial to an efficient and safe operation of the plant. As their facets exhibit surface imperfections and the reflections span distances of hundreds of meters, the flux density distributions generated by a heliostat's reflection on the receiver differ greatly from an assumed ideal. Occlusion caused by shading or blocking, rare misalignment scenarios or extreme flux distributions represent edge cases for flux density prediction. Therefore, each heliostat bears an individual behavior concerning the produced flux density distribution.

To predict this individual behavior, researchers have employed data-driven methods showing an improvement over state-of-the-art approaches by allowing a seamless integration into the calibration process of the camera-target method using the same calibration images as training data.

Their approach shows strong performance, with an error analysis yielding an estimate of 4.5%, where the neural network contributes 1.5%. However, the analysis uses the network's mean training error as a component, leaving edge cases unanswered. As neural networks are uninterpretable models, their integration into safety-critical operations requires thorough uncertainty analysis.

In our work, our aim is to equip the data-driven approach with a layer of security by reasoning about network uncertainties, resulting in an algorithm that filters out certain samples to increase the predictive qualities of the overall process.

1. Introduction

Solar tower power plants are a sustainable source of dispatchable energy that supports the transition toward sustainable energy generation, especially in regions of high solar irradiance. They provide an opportunity not only to generate electricity from incoming sunlight, but also to store energy in the form of heat to make it disposable at night. The plants use a myriad of heliostats, movable mirrors mounted atop a pylon that can track the Sun, to concentrate incoming sunlight onto a receiver on top of a tower.

To efficiently use and convert the incoming irradiation, it is crucial to have a precise knowledge of the flux density distribution that acts on the receiver at any moment. With the rise of Artificial Intelligence (AI) and Machine Learning (ML), data driven methods for flux

density prediction are permeating the research field through various publications [1–4].

Although these approaches yield widespread potential for prediction accuracy, they also require an intelligent approach to data assembly. Kuhl et al. [1] proposed a method to gain training data by using images taken from the calibration target while calibrating the heliostats. This way, there is no additional effort to be made to acquire data for training flux density prediction algorithms. Using a U-Net structure, they developed a possibility to convert grayscale image of single-heliostat reflections on the calibration target into flux density maps. The method proved to be effective, producing only small errors in the predicted flux density.

* Corresponding author.

E-mail address: l.sievers@dlr.de (L.T.E. Sievers).

<https://doi.org/10.1016/j.solener.2025.113739>

Received 28 February 2025; Received in revised form 14 May 2025; Accepted 26 June 2025

Available online 17 July 2025

0038-092X/© 2025 The Authors. Published by Elsevier Ltd on behalf of International Solar Energy Society. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

However, even this effective method behaves in a non-predictable way for a small portion of the dataset. During power plant operation, some faulty predictions do not pose a threat to materials, as errors are assumed to be Gaussian and therefore mitigated by superposition of heliostat reflections. However, this assumption can only hold if we assume that the prediction errors are sufficiently independent of each other. If prediction errors arise from a common underlying mechanism that bears a structural weakness, the assumption of independence cannot be upheld. Thus, if we introduce a machine learning algorithm with too much faulty training data, its performance is naturally limited and cannot be trusted per se.

In our work, we present an approach to mitigating the impeding effects of model inaccuracies by analyzing the inherent neural network uncertainty. We use Monte-Carlo-Dropout to reason about the uncertainty and use a tailored post-processing procedure to feed a binary classification algorithm that decides whether a predicted sample is trustworthy.

The manuscript is structured in the following way: in Section 2 we present the state-of-the-art for flux density prediction methods, Section 3 describes the methods we used in our experimental setup and Section 4 exhibits the experimental setup. In Section 5, we present the experimental results, which are discussed in Section 6. In Section 7 we address further research to be conducted in the realm of uncertainty quantification for AI applications in solar tower power plants.

2. State of the art

2.1. Camera-target method

The state-of-the-art method for heliostat calibration is the camera-target method by Stone [5]. The cornerstone of this method is to aim a single heliostat's focal spot onto a calibration target and compare the aspired center of mass of the flux density distribution to the actual one by the means of digital imaging. By a backward calculation, the errors in the parameters of the geometric model representing the heliostat can be estimated and accounted for.

This method builds the foundation for various calibration approaches. Among others, Pargmann et al. [6], Smith and Ho [7] and Guangyu and Zhongkun [8] improved it to achieve accurate and automated calibration results.

The images taken during the camera-target process can be repurposed to not only derive the heliostat calibration, but also estimate the flux density distribution hitting the calibration target. To achieve accurate information about the flux density acting on the solar receiver, it is useful to know about the position and the shape of the single heliostats' focal spot, as their superposition generates the overall flux density distribution. Therefore, repurposing images that have already been taken is conducive to the efficient and effective operation of a solar tower power plant.

2.2. Flux density prediction

High temporal and spatial gradients in the flux density distribution at the solar receiver pose a risk to the material, as it has to withstand high strains and stresses. Overexploitation of the durability of the material can shorten the lifespan of a power plant, leading to reduced economic and ecological efficiency.

Therefore, the efficient operation of a solar tower power plant requires precise real-time knowledge about the flux density distribution that acts on the solar receiver. As reviewed in [9], there are direct and indirect methods to estimate the flux density distribution. Indirect methods calculate the flux density by assessing the reflections of a Lambertian target that is illuminated by a single heliostat. This approach requires knowledge about the target's reflectivity distribution, including surface irregularities, which cannot be expected in an industrial context.

Kuhl et al. [1] bridge this gap employing a neural network that learns the image-to-image relationship between the images taken during the camera-target process and the respective flux density distribution.

2.3. UNet3+ for flux density prediction

The neural network architecture UNet3+ that is assessed was introduced by Huang et al. [10] to tackle image-to-image-translation tasks. The architecture consists of an encoder stage and a decoder stage, with 5 levels each, encompassing several skip connections that transfer information between different resolutions and levels of abstraction.

Kuhl et al. [1] trained the network on synthetically generated calibration target images with their corresponding flux density maps and background images. The aim of the network is to predict a pair of images, one showing the grayscale background and the other representing the flux density distribution at play. The images that are fed into the network contain 256×256 pixels in a single channel. The outputs of the network are each of the same shape. With this input-output structure and domain knowledge about target background reflectance maps, the algorithm can perform a sanity check during the training process to analyze whether the predicted flux density distribution and background image would produce the grayscale image that was presented to the network as input.

3. Methodology

The objective of this work is to present a binary classification algorithm that is able to identify samples for which the predictive quality of the neural network for flux density deduction is limited. To this end, we implemented a chain of steps for data preparation, feature engineering, and hyperparameter optimization, which we will discuss in this section.

The general outline of our methodology is the following. First, we describe the dataset on which our analyses were performed. Then, we present the metrics by which we will judge the predictive results of the method of Kuhl et al. [1]. We then explain the range of features that our method will draw conclusions from, before we present the mechanism of our classification method. We use a technique for Uncertainty Quantification (UQ) for neural networks called Monte-Carlo-Dropout (MCD) to gather information about the internal uncertainty estimates of neural networks. The hyperparameters of this technique are optimized using a genetic algorithm. The information gathered by MCD and the feature engineering is then fed into a classifier, the details of which we will discuss at the end of this section.

3.1. Dataset

This text summarizes the explanation provided in [1]. The U-Net, as presented by Kuhl et al. [1], is tasked with converting a grayscale image of the calibration target into a corresponding background image, representing its appearance in the absence of a heliostat, and generating a map of the respective flux density distribution at the receiver. The training data is simulated using actual background images captured at Solar Tower Jülich (STJ) and flux density maps calculated by a raytracer, utilizing deflectometry measurements to ensure precision. During inference, data points are randomly sampled from a pool of 896 background images and 1000 fluxes. Following the calculation of the superposition of background and flux density (for further details, see [1]), random scaling is applied to ensure the uniqueness of each data point.

3.2. Consistency metrics

Suitable consistency metrics are crucial to understanding the quality and reliability of a given flux density prediction method. To categorize the predictions into reliable and faulty ones, we employed 6 distinct metrics that quantify different aspects of the quality of flux density prediction.

Let $f : \mathbb{R}^{256 \times 256} \rightarrow \mathbb{R}^{256 \times 256}, x \mapsto y$ be an algorithm that converts a gray scale input image x of the target into a prediction y of the flux density. Let $\hat{y} \in \mathbb{R}^{256 \times 256}$ be a discretization of the true flux density distribution. In Fig. 1 we show some examples of input images (I), flux predictions (X_r), ground truths (T_r) and error maps (E).

For some metrics, we will compartmentalize the images into 32×32 separate blocks of 8×8 pixels, creating some potential confusion in notation. For notational clarity, we define index sets $I_1 = \{1, \dots, 256\}^2$, $I_2 = \{0, \dots, 7\}^2$ and $I_3 = \{0, \dots, 31\}^2$. We classify a prediction by the following metrics:

1. Sum of Block Deviation

$$\sum_{(k,l) \in I_3} \left\| \left(\sum_{(i,j) \in I_2} y_{(k-8+i),(l-8+j)} - \sum_{(i,j) \in I_2} \hat{y}_{(k-8+i),(l-8+j)} \right) \right\|$$

This metric quantifies a segmentation of the domain into 32×32 blocks of 8×8 pixels. It is motivated by the fact that the receiver at the STJ consists of 30×36 cups that act as separate entities with independent physical properties. Although it does not portray the real dimensions exactly, the partition of the input image of 256×256 pixels into blocks of 8×8 pixels is the closest approximation to investigate this behavior.

This metric gives the sum over the absolute flux density deviation of each block. For a reliable flux density prediction we expect the single blocks to exhibit total flux deviations from the reality only within the threshold t_1 .

2. Maximum of Block Deviation

$$\max_{(k,l) \in I_3} \left\| \left(\sum_{(i,j) \in I_2} y_{(k-8+i),(l-8+j)} - \sum_{(i,j) \in I_2} \hat{y}_{(k-8+i),(l-8+j)} \right) \right\|$$

This metric gives the maximum of the per-block deviations. A prediction that deviates from the target value by more than t_2 in a particular block is an indicator for untrustworthiness. This translates to a receiver cup exhibiting significantly different thermal stresses than predicted, which bears a risk to the plant operation.

3. Ratio of Block Deviation

$$\left\{ \frac{(k,l) \in I_3 : \left(\sum_{(i,j) \in I_2} y_{(k-8+i),(l-8+j)} - \sum_{(i,j) \in I_2} \hat{y}_{(k-8+i),(l-8+j)} \right) > \varepsilon \cdot 64}{1024} \right\}$$

This metric encapsulates the ratio of blocks for which the mean deviation surpasses a threshold ε . 1024 is the number of blocks and 64 is the number of pixels per block. We set ε to 0.1. A reliable flux density prediction should not deviate above a threshold t_3 for too many blocks.

4. Ratio of Pixelwise Deviation

$$\frac{\sum_{(i,j) \in I_1} |y_{ij} - \hat{y}_{ij}|}{\sum_{(i,j) \in I_1} \hat{y}_{ij}}$$

This quantifies the error in the flux density prediction in comparison to the total ground truth flux density. The sum of pixelwise errors should stay below a threshold t_4 to be considered a reliable prediction.

5. Ratio of Total Deviation

$$\frac{\sum_{(i,j) \in I_1} y_{ij} - \sum_{(i,j) \in I_1} \hat{y}_{ij}}{\sum_{(i,j) \in I_1} \hat{y}_{ij}}$$

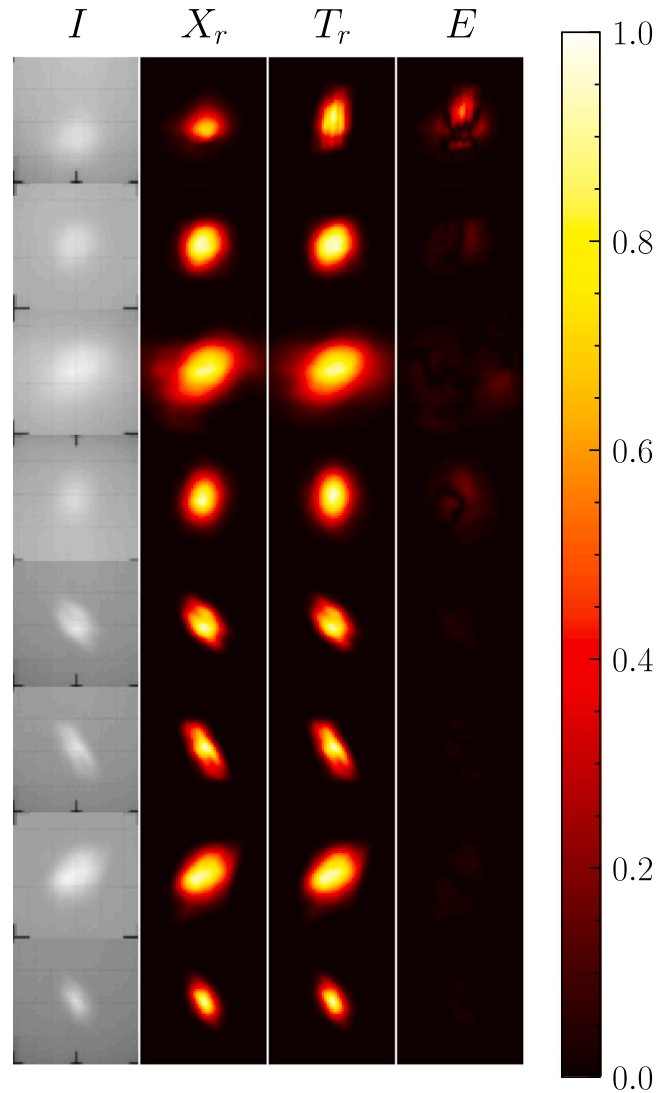


Fig. 1. Examples of predictions with significant deviation from the ground truth (rows 1–4) and of trustworthy predictions (rows 5–8). Here, I denotes the input image to the network, X_r denotes the predicted flux density distribution, T_r the ground truth for the flux density and E the pixelwise absolute error in the prediction.

This expresses the deviation in the total predicted flux density as a percentage of the ground truth. We expect the ratio of the ground truth flux that is an error to stay below a threshold t_5 for the prediction to be trusted.

6. Ratio of Deviating Pixel Values

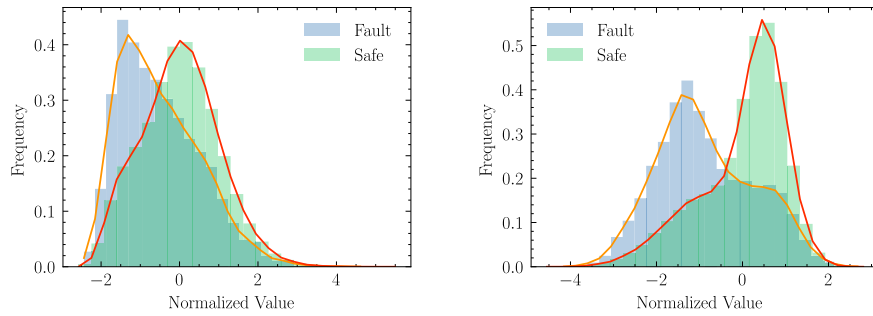
$$\left\{ \frac{(i,j) \in I_1 : |y_{ij} - \hat{y}_{ij}| > \varepsilon}{65536} \right\}$$

This counts the number of pixels for which the prediction is off by an amount larger than the threshold ε . We set ε to 0.1. We expect the ratio of pixels breaking this threshold to stay below t_6 .

We set the thresholds the following way:

$$t_1 = 1000, \quad t_2 = 10, \quad t_3 = 0.01, \quad t_4 = 0.05, \quad t_5 = 0.05, \quad t_6 = 0.1$$

These values are set intuitively and without claim of general applicability. However, they can be altered, resulting in a different classification problem.



(a) This depicts the different distributions for the maximum of the block-wise sum in the kurtosis background images calculated from the MCD outputs.

(b) Here we illustrate the distributions for the kurtosis in the variance image calculated from the MCD outputs.

Fig. 2. Statistics exhibit different distributions depending on the trustworthiness. As there is a massive class imbalance, the numbers for faulty samples are scaled such that the difference in distribution is visible.

3.3. Feature engineering

The eventual binary classification that decides on the trustworthiness of a prediction can draw on different sources of information.

1. The sample itself

Algorithms could draw from properties of the image fed into the neural network to reason about whether a prediction of this sample is expected to be accurate. For example, shading, blocking, or other artifacts that affect the performance of the neural network could be detected.

2. The prediction

The deterministic prediction of the unaltered neural network can be a hint to unusual behavior if its properties vary significantly from the distribution of trustworthy samples.

3. The background prediction

An unrealistic background prediction can also be hint on a faulty flux density prediction as both were part of the training data set.

4. The distribution of MCD predictions.

This is where the information about network uncertainty comes into play. We analyze the distribution of MCD predictions to reason about the trustworthiness. To represent the distribution, calculate the images of per-pixel variance, skew and kurtosis for both the flux image and the background image.

This sums up to 9 different 256×256 arrays of data sources to calculate statistics. For each of the sources of information we calculate several statistics:

- Above Threshold Ratio** This denotes the ratio of pixels for which the values lies above the threshold 0.9.
- Block Max** This represents the maximum of the block-wise sum of values.
- Block Max Variance** This denotes the maximum of the variances in the pixel values for each block. The rest of the statistics are self-explanatory.
- Kurtosis**
- Maximum**
- Minimum**
- Mean**

Calculating each statistic for each of the data sources, we end up with 63 statistics describing a sample. We scaled each statistic to exhibit standard mean and variance across the training set. Fig. 2 exhibits, how faulty and safe samples differ along the dimension of two different normalized statistics.

3.4. Monte-Carlo-Dropout

In general, dropout is a standard technique to combat overfitting in the neural network training process. The approach is to randomly

mask neuron activations (that is, set them to zero) with a probability p to incentivize the network to learn patterns through different neural pathways. This helps the neural network to avoid over-specializing on the presented training data and enables it to recognize the higher-level patterns in the data. During inference, the dropout mechanism is switched, and the neural network can leverage every learned pathway for its decision making.

The idea of MCD [11] is to keep the dropout mechanism turned on during inference. This turns the neural network into a non-deterministic predictor. If we present a test sample to the network more than once, we will get different results every time. The distribution of outputs can then be analyzed to reason about the networks uncertainty. If the neural network is certain about a prediction, we expect it to recognize the same patterns across different pathways, resulting in a low variance of MCD outputs.

For the assessed neural network of Kuhl et al. [1] we identified 5 positions in which we could place dropout layers. To make the most of this uncertainty quantification, we optimized the dropout probabilities p_1, \dots, p_5 .

Each parameter signifies the likelihood of a neuron in the corresponding layer being deactivated. Consequently, a larger parameter indicates a diminished representational capacity of the layer and, by extension, the entire neural network. When a prediction necessitates the majority of the network's informational capacity, a MCD with elevated dropout probabilities is likely to yield a prediction distribution characterized by high variance. Conversely, a less complex prediction will activate multiple pathways within the network, resulting in a more stable prediction when subjected to dropout.

3.5. Genetic algorithm

The genetic meta-algorithm is an optimization technique that can be used for nondifferentiable target functions in a small-dimensional definition space. We used a custom implementation to optimize dropout probabilities $p_1, \dots, p_5 \in [0, 1]$ of the MCD procedure.

The idea of a genetic algorithm is to simulate the evolution of a population that is evaluated regarding the fitness given by a specific function. In each epoch, the individuals of the population are evaluated and ranked by their fitness. The fittest individuals survive into the next epoch while another sub-pool of fit individuals can reproduce by mixing their genes to create new individuals that populate the next generation. Random mutations can change the genes, altering the fitness of the individuals and creating potential for a fitter generation. By iterating over the epochs, the fitness of the population increases and approaches an optimum.

In our work, we regarded the 5 dropout probabilities as the genes making up individuals. Thus we optimized the fitness over the space

$[0, 1]^5$. As the target fitness function we chose the metrics of how well a binary classifier would perform on the dataset that was produced by the MCD algorithm with the given dropout probabilities. Since the evaluation of this fitness function would involve a further training procedure, it would be too costly to evaluate this for every individual in every generation. Therefore we had to rely on a surrogate fitness function hinting at the predictive qualities resulting from the MCD outputs.

We developed a naive surrogate binary classification algorithm that works the following.

1. The MCD algorithm is performed on a dataset of gray-scale images, alongside with standard prediction of the neural network to deduce the output y and the MCD outputs $\{\tilde{y}_i\}_{i=1, \dots, n_{mcd}}$.
2. The per-pixel variances $(v^{(jk)})_{jk} = \left(\text{Var} \left(\left\{ \tilde{y}_i^{(jk)} : i = 1, \dots, n_{mcd} \right\} \right) \right)_{jk}$ are calculated for each sample.
3. The number of pixels, for which the variance exceeds a threshold τ is counted.
4. The decision boundary for the number of pixels breaking the threshold τ is chosen. The idea is that a larger number of pixels with high MCD variance coincides with a larger probability of a faulty prediction. We chose the threshold, such that a binary classification with that threshold maximizes the F1 score on the given dataset.

This optimization algorithm is sufficiently efficient to deliver a surrogate fitness for the genetic optimization.

Later on, we finetune on the binary classification by leveraging feature engineering and random forest classification.

We tested the genetic algorithm on a population size of 10 and for 100 generations. The selection of these hyperparameters was informed by the substantial computational demands associated with the genetic optimization process. Given that the runtime increases linearly with both the population size and the number of generations, the range of feasible options was constrained. As illustrated in Fig. 4, the fitness metric reaches saturation during the training process. Using this method, we found the most promising dropout probabilities to be

$$p_1 = 0.127 \quad p_2 = 0.083 \quad p_3 = 0.86 \quad p_4 = 0.945 \quad p_5 = 0.098. \quad (1)$$

3.6. Classification algorithm

We used the most promising features to feed a Random Forest Classifier (RFC) algorithm. A random forest is an ensemble method that uses multiple decision tree classifiers to predict the class to which a sample belongs. A decision tree is a ML algorithm that uses training data to create decision boundaries on subsets of the features of the given data set.

For our purpose, we used a forest of 100 decision trees. They each have a maximum depth of 10, consider at most 30 features for a single split decision, and uphold a minimum of 5 samples for each leaf node.

3.7. Classification metrics

In this study, we employ various metrics to evaluate the efficacy of a binary classification algorithm.

- **True Positive (tp):** This metric represents the count of instances accurately identified as belonging to the class of faulty samples.
- **True Negative (tn):** This metric indicates the number of instances correctly classified as non-faulty.
- **False Positive (fp):** This refers to the count of instances incorrectly identified as faulty.
- **False Negative (fn):** This denotes the number of faulty instances that are not correctly identified as such.

Table 1

Number of samples falling into different amounts of error categories.

Errors per Sample	0	1	2	3	4	5	6
Samples	289 127	3779	919	362	122	78	13

- **Recall (rec):** This metric quantifies the proportion of faulty samples that are accurately detected.
- **Precision ($prec$):** This represents the proportion of identified faulty samples that are indeed faulty.
- **F1-Score (F_1):** This is the harmonic mean of recall and precision:

$$F_1 = 2 \cdot \frac{rec \cdot prec}{rec + prec} = \frac{2tp}{2tp + fp + fn}$$

The F_1 -Score ensures a balance between recall and precision during optimization, providing a meaningful classification metric even for datasets with significant class imbalance.

4. Experiments

4.1. Genetic algorithm

We leveraged the genetic algorithm using a proxy fitness function on a small dataset to yield a reasonable set of dropout probabilities for the MCD procedure we use for the binary classification. As the genetic algorithm involves many calculations of the fitness function, we restricted the population size to 10 and ran the algorithm for 100 generations.

4.2. Binary classification

As described in [1], the training images are synthesized using a set of preprocessed photographs of the calibration targets as background images and a set of captured flux density distribution maps. The set of background images consists of 896 samples and the flux density distributions span 1000 individual maps. Therefore the total set of available data pairs has a quantity of 896 000.

We conducted the experiments on a data set of 294 400 samples in total. The dataset was split into training, validation and test datasets containing 40%, 30% and 30% of the dataset, respectively. [omment=Rev. 2 Com. 7]The dataset was sampled to ensure a disjoint partition, meaning that each pair of background image and flux density distribution appears no more than once. Each dataset was sampled individually from the total set of available data points. As for each data point generation, the background image and the flux density distribution are sampled randomly and independently, there is a possible overlap between training, validation and test datasets.

With our defined error categories, 5273 of the 294 400 - or 1.79% - would be classified as faulty predictions. Table 1 presents the number of samples exhibiting various types of errors, while Fig. 3 illustrates the distribution of these error types across the dataset.

We trained 25 random forests separately, each of them on a random dataset split to observe the robustness of the classification algorithm with respect to different training data and its order of presence.

5. Results

5.1. Genetic algorithm

As shown in Fig. 4, the Genetic Algorithm managed to raise the surrogate fitness of the population to a satisfactory level, achieving F1 scores of around 60% in the small dataset. This suffices to choose the best performing dropout probabilities for the binary classification mechanism.

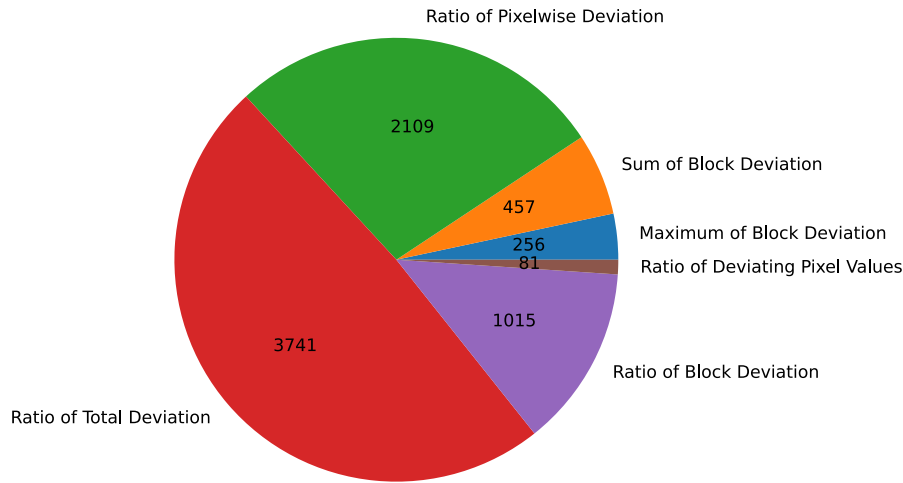


Fig. 3. Each of the potential errors is prevalent in a significant number of samples.

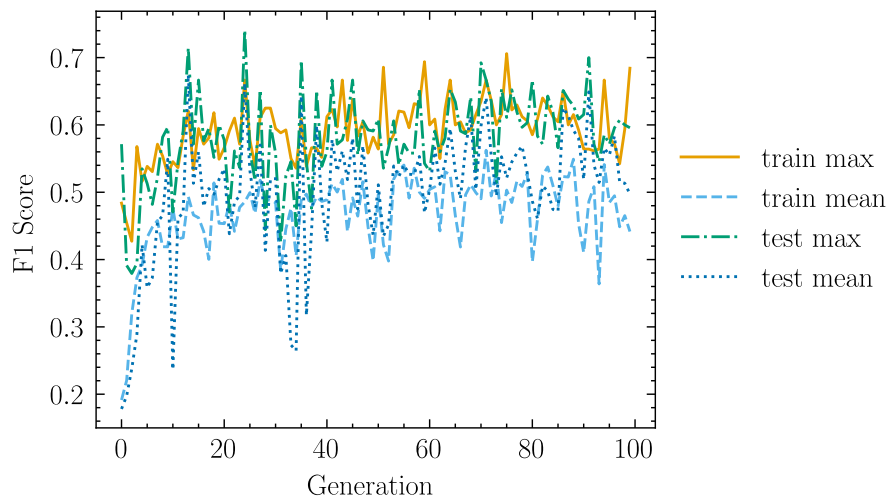


Fig. 4. The F1 curves for the Genetic Algorithm. For the train and the test dataset, the maximum and mean F1 scores attained throughout the population is plotted.

5.2. Classification procedure

The classification results are shown in Fig. 5. The RFCs obtain very stable performance across the different test runs on the independently sampled datasets. Although the UNet algorithm itself is already meticulously trained and very accurate, we can still identify a significant number of samples for which the flux density prediction is faulty. With a precision score of around 38% and a recall score of around 63%, we can reduce the number of faulty predictions by 63%. To achieve that, we only have to reduce the number of samples by $\frac{p_{faulty} \cdot recall}{precision} = \frac{1.79\% \cdot 0.62}{0.36} = 2.97\%$.

In approaches, where the UNet is used as a data generation process for training a ML algorithm, this result is useful, since it yields a cleaner data set while only minimally limiting its extent.

5.3. Feature importance

We conducted an analysis of the fitted random forests to ascertain the features that most significantly influence classification decisions. As illustrated in Table 2, a substantial portion of the importance is concentrated in the top 10 features, which collectively account for approximately 96% of the importance on average. Notably, 6 of these 10 most critical features are derived from the MCD procedure, underscoring the significance of the approach presented. Furthermore, we evaluated the contribution of each data source to the classification

Table 2

Average importances of the features contributing more than 1% of the information used in the decision process of the random forest. The other 52 features contribute 4.26% on average.

Feature	Mean importance
out_bg_above_threshold	11.33%
kurtosis_bg_above_threshold	11.24%
var_flux_above_threshold	10.93%
out_above_threshold	10.52%
sample_above_threshold	10.5%
var_bg_above_threshold	10.1%
kurtosis_flux_above_threshold	9.9%
skew_flux_above_threshold	7.75%
out_mean	2.18%
skew_flux_mean	1.09%
Sum	95.74%

algorithm. It is noteworthy that all data sources contribute relatively equally to the average random forest prediction, as depicted in Table 3.

5.4. Sensitivity analysis

Given that varying operational requirements may necessitate distinct error thresholds, we expanded our examination of the classification procedure through a sensitivity analysis. The computational cost

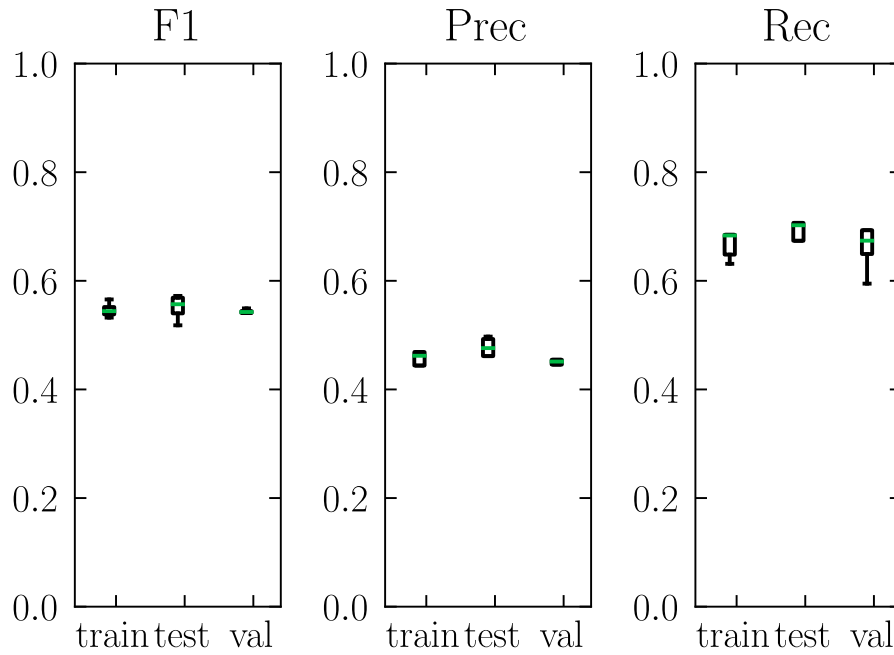


Fig. 5. Classification metrics for the different Random Forest instances on the randomly sampled training, validation and test datasets. The F1 scores for each run and dataset lie consistently above 50%.

Table 3

Each data source exhibits a comparatively equal average category importance. This highlights the usefulness of every source that is used.

Feature	Mean importance
kurtosis_bg	11.26%
kurtosis_flux	11.25%
out	13.34%
out_bg	11.96%
sample	10.75%
skew_bg	10.27%
skew_flux	9.56%
var_bg	10.25%
var_flux	11.36%

constrained our ability to extensively explore the threshold space. Consequently, we conducted a limited grid search, evaluating all threshold configurations that maintained all but one threshold constant while adjusting the other by a factor of 2 or 0.5, respectively. Specifically, for t_1 , we evaluated configurations with $t_1 = 500$ and $t_1 = 2000$. This approach resulted in a total of 13 configurations, allowing us to assess the sensitivity of the classification procedure to each threshold in both directions. The detailed results are presented in Table 4. Depending on the threshold values, F1 scores can reach up to 80%. As error thresholds become more stringent, the positive rate of our algorithm increases. Fig. 6 illustrates that the classification tends to be more accurate as the positive rate rises.

5.5. Decision boundaries

As previously stated, the decision boundaries for the classifiers were selected to maximize the F1-Score on the training set. However, in certain circumstances, operators may not wish to maintain a balance between precision and recall. If one of the two potential errors, namely false positives or false negatives, poses a greater risk to the operation, operators may prioritize maximizing one of the metrics while improving the other. The selection of the decision boundary influences the relationship between recall and precision. Fig. 7 presents an example of a Precision–Recall curve for a classifier that demonstrated exceptional

Table 4

The results of the sensitivity analysis. Each experiment resulted in a F1 score of at least 45%, emphasizing the robustness of our approach. For some configurations, even F1 scores of 80% can be reached.

Configuration	F1	Mean FP	Mean FN	Mean TP
normal	54.68%	0.40%	0.99%	0.84%
$t_1 = 2000$	54.12%	0.45%	0.98%	0.84%
$t_1 = 500$	80.49%	0.94%	1.69%	5.41%
$t_2 = 20$	54.08%	0.42%	0.99%	0.83%
$t_2 = 5$	59.11%	1.07%	2.23%	2.38%
$t_3 = 0.02$	54.23%	0.47%	0.94%	0.84%
$t_3 = 0.005$	56.89%	0.47%	0.98%	0.95%
$t_4 = 0.2$	45.73%	0.74%	0.88%	0.68%
$t_4 = 0.05$	76.11%	13.05%	8.71%	34.65%
$t_5 = 0.1$	71.19%	0.16%	0.29%	0.55%
$t_5 = 0.025$	47.54%	4.72%	5.11%	4.45%
$t_6 = 0.1$	54.02%	0.49%	0.97%	0.86%
$t_6 = 0.025$	54.29%	0.49%	0.96%	0.86%

performance. In this context, precision can reach as high as 44% while maintaining a recall of 100%. If false positives are considered a significant error, this configuration allows the classifier to identify 44% of the errors without triggering a false alarm.

6. Discussion

Our analysis of the neural network employed in the flux density prediction exhibited information about the prediction quality hidden in MCD layer activations. In our work, we could use this quantification of uncertainty to reason about the predictive quality of samples during the inference of an effective neural network, thus significantly increasing the reliability.

Our research shows that even for this very performant network we can find additional information to its mere predictions that can signify whether the prediction is trustworthy or not. We found a reliable algorithm that can filter out faulty predictions with a high accuracy, providing the possibility to fall back to classical algorithms or eliminate the sample altogether.

While an incorrect prediction of the flux density of an individual heliostat does not significantly impact operational efficiency or

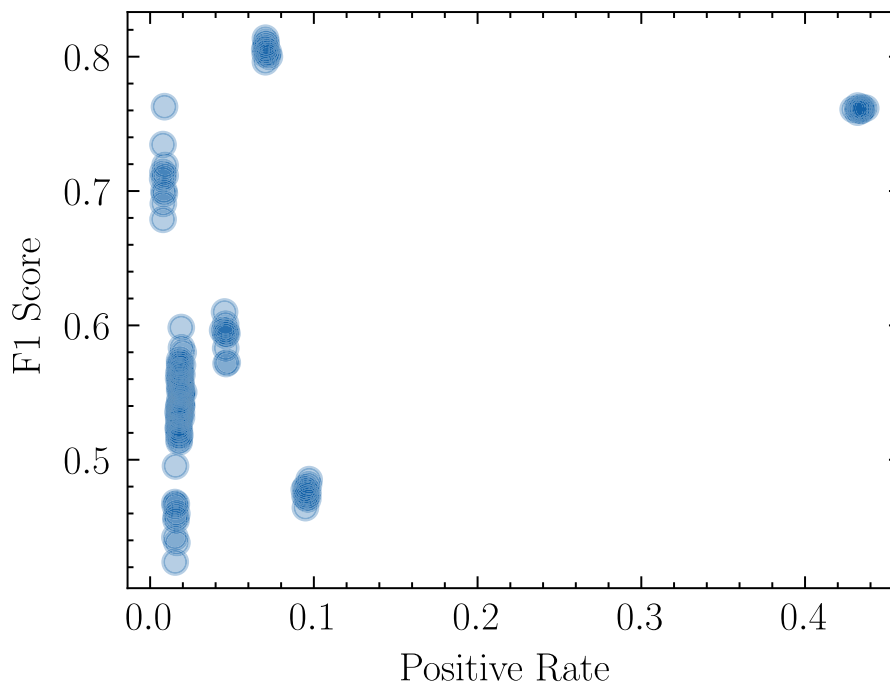


Fig. 6. This scatter plot depicts the relationship between the positive rate, i.e. the ratio of faulty samples, and the achieved classifier accuracy. Those classification scenarios that produce a high positive rate result in a classifier with high F1 score.

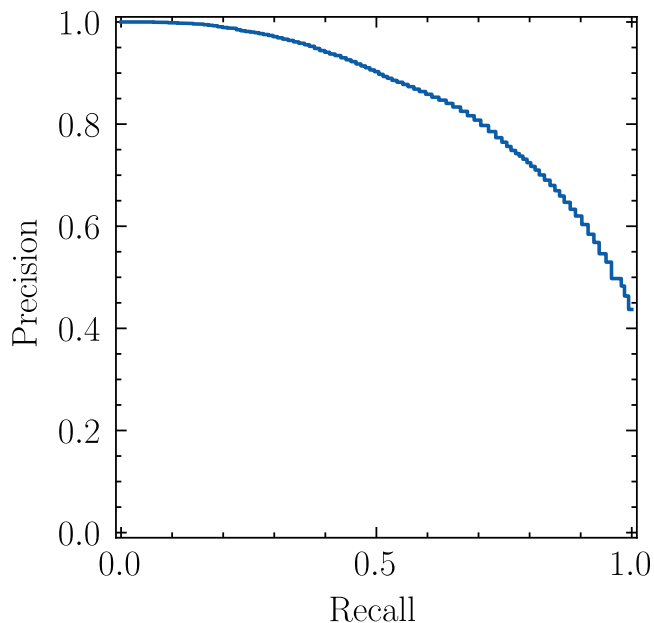


Fig. 7. Precision–Recall curve for an estimator with high F1-Score.

safety, the presence of false negatives is less critical compared to other fault detection scenarios. Nevertheless, a high proportion of erroneous predictions can lead to substantial inaccuracies in the overall flux density distribution on the receiver, thereby threatening both efficiency and safety. Conversely, false positives merely necessitate reverting to a more reliable, albeit potentially slower, algorithm, resulting in increased runtime.

Our approach is highly applicable to all types of neural network models that play a role in solar tower power plants. With sufficient knowledge of the data domain, it can be customized to be a tool for any prediction method that is part of decision making in high-stakes scenarios.

In addition to the highly effective algorithm presented by Kuhl et al. [1], our work can serve as a security layer that is an indicator of the reliability of every prediction. Although for faulty examples, our algorithm cannot correct the predictions, it can identify them. Our analysis of decision boundaries demonstrates its adaptability to the specific requirements of industrial operations. This adaptability facilitates the integration of decision rules and boundaries, as well as the application-specific calibration of false positives and false negatives. Further decision making would lie in the responsibility of the operators. Their options include falling back to a stable but potentially more cost-intensive or error-prone approach or neglecting the sample altogether.

7. Outlook

In the case of the network presented, our work could help creating a cleaner dataset, as the U-Net structure is supposed to be used as a data generation process for other ML techniques as well.

On top of that, our approach is highly applicable and customizable to other ML applications in the realm of solar tower power plants. The use of UQ information to reason about the trustworthiness of flux density predictions is merely an example of the possibilities that arise from the additional potential of neural networks. Once a neural network is trained and proves to be effective, an MCD mechanism is easily installed into the structure. Our approach to streamline the MCD outputs into a binary classification algorithm can be followed even in other applications. Of course, for a different context, the design choices would have to be altered accordingly. This includes the rules by which to discard predictions, the features to extract from the samples and predictions or even the choice of classification algorithm and proxy fitness function in the Genetic Algorithm.

When regarding the problem of UNet-supported flux density prediction, our approach can be finetuned to the respective requirements for the predictions at each individual solar tower, simply by redefining the rules that classify the predictions into faulty and trustworthy ones.

We presented a further step towards reliable data-driven flux density prediction at solar tower power plants. With our addition to the approach of Kuhl et al. [1] we can expand on the reliability of the UNet

approach as a data generation process for the training of ML algorithms or a stand-alone solution to flux density prediction on calibration targets. Our analysis makes assumptions about which errors in the flux density predictions are tolerable and which are not. As these decisions were made intuitively, it would be interesting to assess the performance of our approach under realistic conditions, i.e. tuning the thresholds to the values that practitioners deem appropriate for a specific solar tower.

Moreover, an adoption of our binary classification of neural network predictions to other process steps in the operation of solar tower power plants would be insightful. The method we developed could act as a blueprint for providing AI approaches in high-stakes applications with a layer of security to facilitate their adoption in an industrial context.

List of abbreviations

AI Artificial Intelligence

ML Machine Learning

STJ Solar Tower Jülich

MCD Monte-Carlo-Dropout

RFC Random Forest Classifier

UQ Uncertainty Quantification

CRediT authorship contribution statement

Leon Tim Engelbert Sievers: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation. **Daniel Maldonado Quinto:** Writing – review & editing, Supervision, Project administration, Funding acquisition. **Bernhard Hoffschmidt:** Supervision, Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

- [1] M. Kuhl, M. Pargmann, M. Cherti, J. Jitsev, D. Maldonado Quinto, R. Pitz-Paal, In-situ UNet-based heliostat beam characterization method for precise flux calculation using the camera-target method, *Sol. Energy* 279 (2024) 112811, <http://dx.doi.org/10.1016/j.solener.2024.112811>, URL: <https://www.sciencedirect.com/science/article/pii/S0038092X24005061>.
- [2] M. Kuhl, M. Pargmann, M. Cherti, J. Jitsev, D. Maldonado Quinto, R. Pitz-Paal, Flux density distribution forecasting in concentrated solar tower plants: A data-driven approach, *Sol. Energy* 282 (2024) 112894, <http://dx.doi.org/10.1016/j.solener.2024.112894>, URL: <https://www.sciencedirect.com/science/article/pii/S0038092X24005899>.
- [3] X. Lin, X. Zhao, Z. Liu, W. Huang, Y. Zhao, J. Feng, Real-time and high-accuracy radiative flux distribution simulation based on analytical model for solar power tower system, *Sol. Energy* 287 (2025) 113208, <http://dx.doi.org/10.1016/j.solener.2024.113208>, URL: <https://www.sciencedirect.com/science/article/pii/S0038092X24009034>.
- [4] F.J. Collado, One-point fitting of the flux density produced by a heliostat, *Sol. Energy* 84 (4) (2010) 673–684, <http://dx.doi.org/10.1016/j.solener.2010.01.019>, URL: <https://www.sciencedirect.com/science/article/pii/S0038092X10000320>, International Conference CISBAT 2007.
- [5] K.W. Stone, Automatic heliostat track alignment method, 1986, US4564275A.
- [6] M. Pargmann, M. Leibauer, V. Nettelroth, D. Maldonado Quinto, R. Pitz-Paal, Enhancing heliostat calibration on low data by fusing robotic rigid body kinematics with neural networks, *Sol. Energy* 264 (2023) <http://dx.doi.org/10.1016/j.solener.2023.111962>, URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85172320748&doi=10.1016%2fj.solener.2023.111962&partnerID=40&md5=b77de401ba10deafd24bf018f2cb35fe>, Cited by: 0; All Open Access, Green Open Access.
- [7] E. Smith, C. Ho, Field demonstration of an automated heliostat tracking correction method, *Energy Procedia* 49 (2014) 2201–2210, <http://dx.doi.org/10.1016/j.egypro.2014.03.233>, URL: <https://www.sciencedirect.com/science/article/pii/S1876610214006870>, Proceedings of the SolarPACES 2013 International Conference.
- [8] L. Guangyu, C. Zhongkun, Heliostat attitude angle detection method based on BP neural network, *MATEC Web Conf.* 139 (2017) 00043, <http://dx.doi.org/10.1051/mateconf/201713900043>.
- [9] M. Röger, P. Herrmann, S. Ulmer, M. Ebert, C. Prah, F. Göhring, Techniques to measure solar flux density distribution on large-scale receivers, *J. Sol. Energy Eng. Trans. ASME* 136 (3) (2014) <http://dx.doi.org/10.1115/1.4027261>, URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84899844133&doi=10.1115%2f1.4027261&partnerID=40&md5=a5802b730e11559c570fa127d67f073d>, Cited by: 46; All Open Access, Green Open Access.
- [10] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, J. Wu, UNet 3+: A full-scale connected unet for medical image segmentation, in: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2020, pp. 1055–1059, <http://dx.doi.org/10.1109/ICASSP40776.2020.9053405>.
- [11] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, in: M.F. Balcan, K.Q. Weinberger (Eds.), Proceedings of the 33rd International Conference on Machine Learning, in: Proceedings of Machine Learning Research, 48, PMLR, New York, New York, USA, 2016, pp. 1050–1059, URL: <https://proceedings.mlr.press/v48/gal16.html>.