# FlowFrontNet: Improving Carbon Composite Manufacturing with CNNs

Simon Stieber[0000−0002−3753−8264] (✉), Niklas Schröter[0000−0002−7154−3374], Alexander Schiendorfer[0000−0002−5283−5304], Alwin Hoffmann[0000−0002−5123−3918], and Wolfgang Reif[0000−0002−4086−0043]

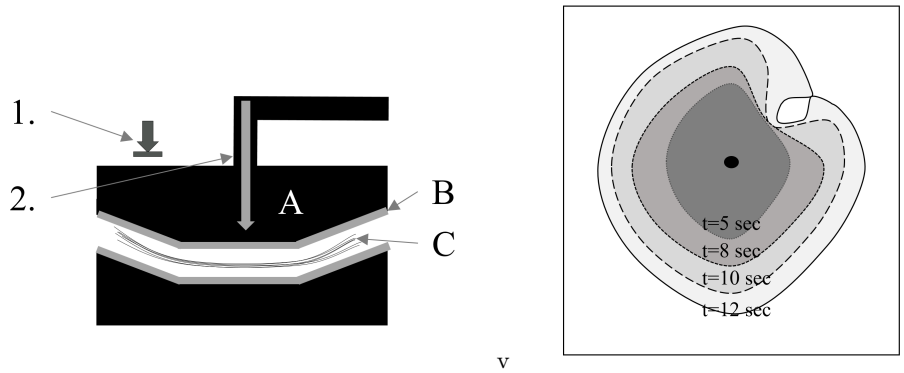
Institute for Software & Systems Engineering, University of Augsburg, Universitätsstr. 6a, 86159 Augsburg, Germany
{stieber,schiendorfer,hoffmann,reif}@isse.de, niklas.schroeter@student.uni-augsburg.de

**Abstract.** Carbon fiber reinforced polymers (CFRP) are light yet strong composite materials designed to reduce the weight of aerospace or automotive components – contributing to reduced emissions. Resin transfer molding (RTM) is a manufacturing process for CFRP that can be scaled up to industrial-sized production. It is prone to errors such as voids or dry spots, resulting in high rejection rates and costs. At runtime, only limited in-process information can be made available for diagnostic insight via a grid of pressure sensors. We propose FlowFrontNet, a deep learning approach to enhance the in-situ process perspective by learning a mapping from sensors to flow front "images" (using upscaling layers), to capture spatial irregularities in the flow front to predict dry spots (using convolutional layers). On simulated data of 6 million single time steps resulting from 36k injection processes, we achieve a time step accuracy of 91.7% when using a $38 \times 30$ sensor grid with 1 cm sensor distance in x- and y-direction. On a sensor grid of $10 \times 8$, with a sensor distance of 4 cm, we achieve 83.7% accuracy. In both settings, FlowFrontNet provides a significant advantage over direct end-to-end learning models.

**Keywords:** Process monitoring, convolutional neural networks, digital twin, manufacturing, industrial automation, resin transfer molding, carbon composites

## 1 Introduction to Composite Manufacturing via RTM

Carbon fiber reinforced polymers (CFRP) are extremely strong composite materials despite their low weight. That makes them attractive for the construction of lighter aerospace and automotive parts (conventionally made from steel or aluminum) to reduce fuel consumption and CO2 emissions [5]. In essence, these composites are made from a so-called polymer matrix that is reinforced with textiles containing carbon fibers. To produce CFRP parts industrially, resin transfer molding (RTM, [1]) is a commonly applied manufacturing process for medium volumes (1,000s to 10,000s of parts) and is depicted in Figure 1a): A liquid thermoset polymer (called a resin) is injected under pressure into a mold cavity that




(a) Front view of the hardware involved in RTM.

(b) Top view of the evolving flow front inside (B), with a dry spot.

**Fig. 1.** Overview of resin transfer molding (RTM): The resin (2.) is injected into a mold cavity (B) filled with textiles containing carbon fibers (C). A press (A) applies the necessary pressure (1.).

contains reinforcement material such as textiles with carbon fibers. This results in a "flow front" that separates impregnated material from dry material, shown in Figure 1b. Thermoset resins are converted from a liquid to a solid through heat – they are "cured".

During the RTM process, several errors can render the result useless and, thus, make the overall production expensive [17, 4, 9]. They occur, inter alia, due to high input variances of the fiber contents in the textile (the "preform"). *Dry spots* refer to areas of the preform that are not impregnated by the liquid, as shown in the top right of Figure 1b. In some cases, these dry spots irreparably invalidate the stability and stiffness properties required for the manufactured part. In others, they can be repaired manually. Either way, automated process monitoring based on sensors applied to the mold would significantly improve the quality assurance – called in-situ monitoring. These sensors (e.g., temperature, dielectric, ultrasound, or pressure) are able to track the flow front of the fluid and, consequently, predict *spatial* deviations from proper RTM runs. This may indicate problems such as dry spots or voids (i.e., enclosures where air is trapped), providing diagnostic insight, or even control actions to avoid rejects.

In this paper, we propose to use machine learning, in particular convolutional neural networks (CNN), to get a binary classifier $f : \mathbb{R}^n \to \{0, 1\}$ from $n$ mold-integrated sensors to labels describing whether, at a given point in time, there is a dry spot present or not. As an intermediate step, we train the network to generate higher-resolution images of the flow front (extracted from an RTM simulation in PAM-RTM[1]) from sensor data. These images capture spatial irregularities in the

[1] https://www.esi-group.com/software-solutions/virtual-manufacturing/composites/pam-composites/pam-rtm-composites-molding-simulation-software

flow. Since such full images are only available during simulation or specialized permeability studies [3] and not in real-world closed molds, a trained model could substantially enhance the spatial information transmitted by actual flow front sensors in productive settings – in the sense of a digital twin [8]. In this paper, we focus on detecting dry spots from sensor input obtained from simulated data as a first step towards a transfer to real data. These are major quality concerns during the injection process [7].

Our model, *FlowFrontNet*, first uses several deconvolutional and convolutional layers to create the image representation from pressure sensor data and proceeds to perform the binary classification using convolutional and dense layers. The overall classification is performed individually for every frame of a simulated injection sequence such that a warning of dry spots can be issued at any time. Our central scientific question is if we can detect dry spots from simulated sensor data and whether the spatial flow-front information captured in convolutional layers improves the classification. We compare the accuracy the simulation-enhanced FlowFrontNet achieves to that of a standard feed-forward network (both based on sensor data) and find major improvements in Section 4. Our goal is to offer FlowFrontNet as a starting point for future research in composite manufacturing: code[2], checkpoints [19] and data [20] are available.

Following a discussion of related work, we present the data regime including input variation for simulation and automated label acquisition in Section 2, present the neural network models and training methodology in Section 3, and conclude with experimental results in Section 4.

### 1.1 Related Work

To cover the relevant related work, two aspects have to be addressed: The technical process and the machine learning models.

There have been several publications on in-situ monitoring the RTM process. Pantelelis et al. [12] present a system on how to detect the curing state but only mention ML-based analyses as future work. Zhang et al. [22] use simulated and actual sensor data to detect the curing rate of the process. They use aggregated shallow neural networks to achieve this based on two sensors. They further use the model to adapt the heating of the process. Our work, by contrast, focuses on the resin injection and detect dry spots in the flow front from multiple sensors. Leveraging the spatial capabilities inherent to CNNs has never been applied to dry spot detection in the RTM process before.

Unrelated to composite production, deconvolutional layers that increase the spatial dimensions have been used for image enhancement tasks. Xu et al. [21] used them to deblur images. Shi et al. [15] approach superresolution, the task of upscaling pictures from low to high resolution, with deconvolutional networks. For semantic segmentation, Noh et al. [11] used a combination of convolutional and deconvolutional networks. They compress the input image with a VGG-16 to afterward decompress the encoding with deconvolutional layers.

[2] https://github.com/isse-augsburg/rtm-predictions

Out of these approaches, [15] comes closest to our task of extracting information from a grid of sensors since the compressed data is known before and not a learned encoding as in [11]. However, our approach works with more heterogeneous data in that it generates images from sensors rather than improving the resolution of conventional photos. Furthermore, the dimensions of the input are enlarged and not kept the same as in [21].

## 2 Creating Training Data from Simulation

To train a detector for dry spots from RTM sensor data, we need to observe sufficiently many training instances. Generally speaking, industrial ML use cases based on sensor data from actual production cycles tend to suffer from limited data quality and quantity. Moreover, setting up a new process (including design of the mold, choice of the resin, etc.) takes time until sufficiently many training runs have been processed, not to mention the material costs. Simulation is a proven remedy for the lack of data [13] which is our focus in this paper. Basic RTM processes are well supported by existing engineering tools [3].

The lack of real high-quality data is only one reason to opt for simulation. Another more pressing reason is that we can create a spatial representation of the flow front, i.e., a "flow front image" (see Figure 3a), that is not observable in real closed molds. Those images will serve as the target for the generative Deconv/Conv part of FlowFrontNet, described in Section 3.

Being able to simulate the process might raise the question of why one has to apply machine learning at all – instead of just running the simulation online. First, the trained models will encapsulate only those aspects of the simulation that are needed to make good dry spot predictions for a given RTM setup. An online simulation would take much longer and be infeasible for real-time monitoring. Second, we can reasonably anticipate that real runs will contain aspects that are not properly captured by simulation. For instance, variability in the process parameters such as the textile might not be observable. Using an ML-model will eventually enable us to add real data to the training.

In the following, we describe our process of obtaining enough (6 million) domain-specific training instances from simulations executed with randomized initial conditions (variances in the input textile) and automatically deriving the corresponding dry spot labels that are images with the filling level as intensities.

### 2.1 Simulated RTM Runs in PAM-RTM

PAM-RTM is a software package designed for laying out RTM processes, including fluid dynamics simulation. Modeling flow transport in porous media mathematically is most commonly carried out based on Darcy's law [2] for one-dimensional flow:

$$v_x = -\frac{k_x}{\eta}\frac{\Delta P}{\Delta x} \tag{1}$$

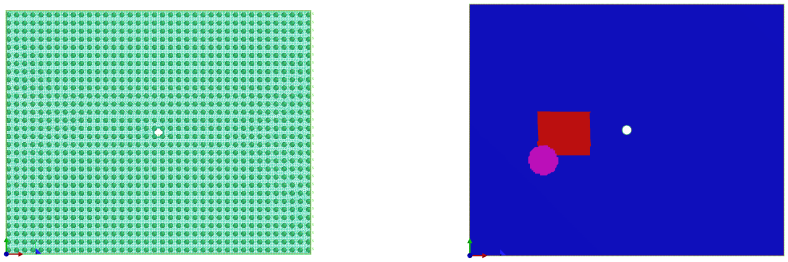


**Fig. 2.** Simulated composite plate with a full sensor grid and central injection point - 1140 sensors located at distance 1 cm in $x$ and $y$. On the right, a fiber volume content (FVC) map is given with local perturbations (rectangular and circular).

where $v_x$ is the 1D flow velocity, $k_x$ represents the permeability value of the textile (corresponding to how "easily" fluids can permeate it), $\eta$ denotes the fluid viscosity, and $\frac{\Delta P}{\Delta x}$ expresses the pressure drop along a specific flow length [3].

PAM-RTM includes other features as well, including draping simulation, the meshing of CAD parts, and distortion during curing which makes it a standard tool in composite manufacturing. For this paper, we only needed the fluid simulation part modeling resin injection. Getting the flow front prediction and, subsequently, dry spot classification right is a high-impact quality goal.

Setting up a particular RTM simulation consists of defining the geometrical 3D-model of the part to be constructed, the viscosity and permeability parameters of the resin and textile, respectively, as well as temperature and pressure of the injection. A single simulation *run* then represents a whole injection that continues to pump resin until the mold cavity is filled entirely.

As our running example, we choose a simple rectangular composite plate with dimensions of 38 $\times$ 30 cm. Adding a simulated pressure sensor for every centimeter in $x$ and $y$ direction yields a total of 1140 sensors, as **??** shows. The sensors can be placed at any location in the simulated composite plate, independent of the mesh grid of the plate. To ensure that the chosen parameters are realistic, this configuration follows the setup for experimental permeability characterization available at the Processing of Composites Group of Montanuniversität Leoben as presented by Grössing et al. [3]. For the textile, the setup assumes a natural fiber fabric with a fiber volume content (FVC) of 0.268 and a 3.5 mm thickness. Finally, the resin used in the model experiment in Leoben was a plant oil with a viscosity of $0.065 \frac{N \cdot s}{m^2}$.

In reality, dry spots emerge from local variations in the textiles (thicker or thinner areas) that lead to regions of low or high fluid permeability, respectively. To recreate that effect in our simulation and obtain varied training data, we altered the fiber volume content in certain areas, as shown in **??**. These perturbations are responsible for the distribution our training and test RTM runs are drawn from. For simplicity, every perturbation corresponds to one circle with a random diameter and one rectangle with random side lengths – with varying strengths of FVC perturbation. All of these perturbations were automatically

**Table 1.** Data set sizes – train/val/test-split

| Data set | # Runs | # Samples | % (Samples) |
|---|---|---|---|
| Training | 29,663 | 5,067,352 | 81.11 |
| Validation | 756 | 131,072 | 2.09 |
| Test | 6,244 | 1,048,576 | 16.78 |
| All | 36,663 | 6,247,000 | 100 |

created for 40k simulation setups that were run in parallel on 10 hosts with 32 cores each. We repeated the simulations with different random strengths of FVC perturbations: setting them to 0.1 - 0.8 covers a wide range of cases (including mild cases that will eventually not produce dry spots), increasing them to 0.2 - 0.8 or even 0.3 - 0.8 provokes more severe perturbations leading to dry spots.

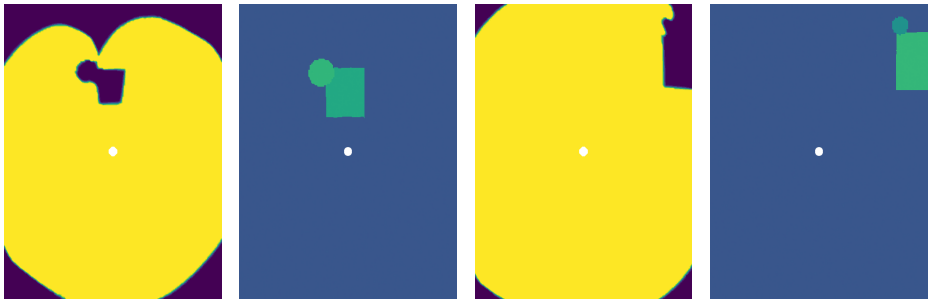
For training our models, Table 1 shows how we split the data obtained from those FVC-perturbed simulation runs. Although we consider individual frames (corresponding to time steps of runs), the splits do not break up runs, i.e., a run is fully contained in either train, validation, or test set. That allows for reusing the data when considering sequence models in the future.

When working with simulations, we have to double-check the time series resolution. We noted that PAM-RTM distinguishes between multi-state results for the simulated pressure sensors and single-state results for the simulation results at every node of the mesh of the plate such as filling status and pressure. While multi-state results are present for every simulated time step (including the respective simulation times), single state results are saved every $k$-th second in simulation time or every $i$-th simulation step. To keep the simulation efforts tractable and produce enough runs, we selected an approximate time resolution of 0.5 seconds and dispose of all pressure sensor results that do not correspond to time steps with available filling status. Finally, before feeding the pressure sensors' values to the ML-models, we divide them by $10^5$ to obtain numerically well-behaved training dynamics in the early layers of the neural networks.

### 2.2 Dry Spot Label Creation

Varying the FVC locally in the input textiles and recording the simulated runs provides us with pressure value time series and flow front developments as "images" where every pixel corresponds to the filling level at a given point in time. To classify dry spots in a supervised manner, we also need labels indicating whether a flow front image contains a dry spot (e.g., Figure 3a) or not (e.g., Figure 3c). And we need those labels for all 6 million frames from 36k runs.

An area counts as a dry spot if it is a non-filled area that is enclosed by resin. Comparing Figure 3a and Figure 3b, we might be tempted to directly derive binary labels from the modified FVC maps that served as simulation input. However, the matter is more complicated. First, an observed dry spot tends to be smaller than the original perturbation since the outskirt still gets



(a) Interior dry spot (b) FVC map corresponding to 3a (c) No dry spot at the flow front (d) FVC map corresponding to 3c

**Fig. 3.** Exemplary flow fronts resulting from locally perturbed FVC-maps extracted from specific time steps

permeated by resin. Second, an FVC perturbation would be the same for all frames of a run even though the dry spot only becomes apparent after the flow front reached it geometrically. Third, a regular flow front might be jagged such as the one in Figure 3c. Such areas should not be counted (yet) as dry spots since they tend to eventually be filled. Thus, we devised a heuristic that combines the perturbed FVC-maps with computer vision techniques on the flow front images to output label maps indicating if a pixel belongs to a dry spot or not.[3]

First, we extract the already filled pixels from each frame, e.g., the light (yellow) pixels in Figure 3a. The negative of that image only contains the dry parts. Since dry spots cannot lie within the filled regions, we focus on the latter for finding contours. We use OpenCV[4] to find contours in those negative images. The contours are constrained to be smaller than the whole flow front but at least larger than the central injection point. However, not all of the contours identified as dry spot candidates must be dry spots. Instead, they could emerge from jagged flow fronts as apparent in Figure 3c) that are not enclosed by resin. To distinguish between these two cases, we enhance the pure computer vision operations (such as contour or hole detection) with the perturbed FVC maps. Since a high FVC corresponds to low permeability and vice versa, we overlap the dry spot candidate contours with the FVC map. For every contour in the dry area, its probability of being a dry spot is estimated proportionally to the percentage of the overlap with an FVC perturbation. To reduce the number of incorrectly labeled dry spots, we look at runs, i.e., sequences of frames, and discard candidates that only occur in a single frame. This is justified since the flow front is expanding continuously and, thus, the probability of a dry spot has to be similar for multiple frames in a run. Some edge cases (e.g., simulation errors or dry spots in areas other than

[3] Note that we did not use the "air entrapment" feature in PAM-RTM since that would prematurely end simulation runs, produces lagging information, and cause the experimental setup to diverge from the model setup in Leoben.

[4] https://opencv.org/

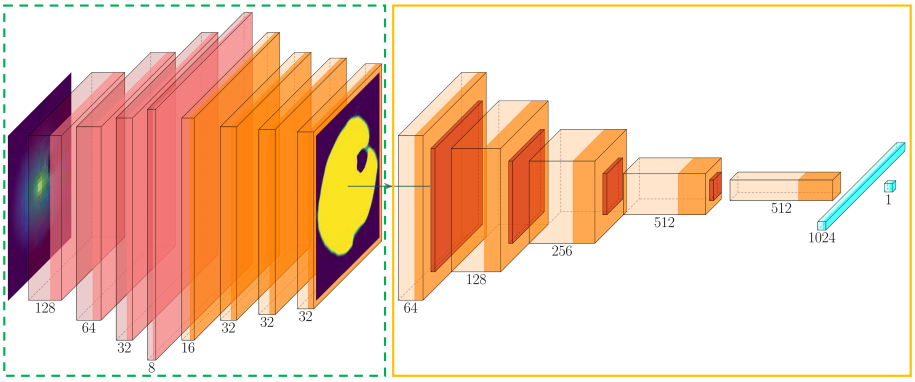



**Fig. 4.** FlowFrontNet: The first part (dashed, green) is a Deconv/Conv network that maps from sensors to images. The second part (solid, orange) is the DrySpotNet consisting of five convolutional/max-pooling layers and two fully-connected layers to perform the final classification task. Numbers denote the resulting feature maps.

the FVC perturbations) lead to unexpected jumps in dry spot probabilities over consecutive frames. To maintain high-quality data, we excluded whole runs with such phenomena – approximately 9.16% of all runs which leaves us with 36,663 valid runs (cf. Table 1).

## 3 Approach - Model and Training

After generating data for different FVC contents in a sufficient amount and labeling them in an automated manner, the next step is to present *FlowFrontNet*, the main model of our approach, which is shown in Figure 4. Before going into detail, we present the key points of this network.

The overall FlowFrontNet maps pressure sensor inputs to classification decisions. It consists of the generative part that upsamples from sensor grids to flow front images (called Deconv/Conv) and the binary classification part (called DrySpotNet). By learning to produce a flow front from sensor input, the network learns a representation of fluid dynamics. The Deconv/Conv part itself is useful for other use cases down the road, say, exact dry spot localization or pixel-wise flow front detection. The later DrySpotNet performs a binary classification on the generated images (see Figures 5b and 5c for example inputs to that part). By adding spatial fluid dynamics knowledge, we aim to surpass the performance of a conventional feed-forward classifier achieves based on the same sensor input.

We present the version of FlowFrontNet that is used for 1140 sensors. The other sensor resolutions (see Table 2) only require slight changes in kernel size and layer count. Those are necessary since the smaller input sizes lead to smaller outputs with the hyperparameters used for the 1140-sensors Deconv/Conv network. Due to the resulting poor image resolution, the results would not be ad-

equate for detecting dry spots. In the following, we describe the intricacies of each network and their training processes.

*Deconv/Conv Network: Sensor Data to Flow Front Images* The first part of FlowFrontNet, *Deconv/Conv*, is a fully convolutional neural network [14] (see Figure 4, left part) that receives the low-resolution sensor grid values and outputs a high-resolution flow front image. The first four layers are deconvolutional to extract features from the sensor array. As opposed to convolutional layers, deconvolutional layers increase the spatial dimensions of an image, known from image segmentation or superresolution [11, 15].

Our approach is similar in this regard: the pressure sensor grid has a low resolution and constitutes a compressed representation of the flow front. We apply deconvolutional layers to increase the resolution of the sensor array while simultaneously filling the spaces between the sensors, i.e., interpolating missing values. Afterward, we utilized five standard convolutional layers (providing the required amount of non-linearity) to create and shape the final flow front image (e.g., with spatial dimensions $117 \times 149$).

The first step is to pre-train the Deconv/Conv network to produce images of the flow front from sensors, as shown in the left part of Figure 4. As mentioned before, this pre-training needs simulated data since flow fronts are hidden in real-world closed molds. All hidden layers are followed by the rectified linear unit (ReLU) activation, while the output layer is activated by the sigmoid function to make sure the generated flow front images lie within the range $[0, 1]$.

*DrySpotNet: Flow Front Image to Binary Dryspot Classification* The second part of FlowFrontNet, DrySpotNet, receives the generated images and classifies them concerning dry spots. The dry spot labels are obtained as described in Section 2.2. The architecture for this classification follows a standard convolutional classification network [6] (see the latter part of Figure 4). The final output yields soft classification scores using the sigmoid function (commonly interpreted as class probabilities) that are eventually thresholded to achieve a hard classification.

*FlowFrontNet: Training End-to-End Sensor Data to Dryspot* After pretraining the Deconv/Conv net using flow front images, we append the untrained DrySpotNet for the final classification. To avoid changing the already trained weights of the generative Deconv/Conv layers, the pre-trained layers are "frozen", meaning that their weights are not updated during training. This is identical to the practice of freezing convolution layers in fine-tuning for special purpose image classification [16]. After training the newly appended output layer, it can be useful to also "unfreeze" the pre-trained weights of early hidden layers during backpropagation. The data is exchanged and parts of the original network are used as the backbone for a new image processing task. Here, our approach is different: we change the objective and targets from generating images to that of doing dry spot binary classification, leaving the early layers fixed.

**Table 2.** Sensor layouts

| Sensors | Layout | Sensor distance - x and y |
|---|---|---|
| 1140 | 38 × 30 | 1 cm |
| 80 | 10 × 8 | 4 cm |
| 20 | 5 × 4 | 8 cm |

When combining the generative part of the Deconv/Conv net with the DrySpot-Net, low sensor input numbers may lead to comparably "blurry" flow front images (e.g., Figures 5b to 5d) before handing them to the classification part. To obtain higher-contrast flow front images and, consequently, better classification results, we add a non-differentiable hard *pixel-threshold* to the forward pass of the network after the Deconv/Conv net. This operation sets all values below the threshold to 0 and all above to 1. Since the flow-front-generating layers of the networks that lie before the pixel-threshold are frozen and need no gradients, we can easily incorporate this operation.

*Feed-forward Network: Baseline Classifier* To judge the merits of FlowFrontNet, we design a basic end-to-end feed-forward network as a baseline classifier. It consists of two ReLU-activated, fully-connected hidden layers and a sigmoid output for the dry-spot probabilities. As input, it receives the sensor values described in Section 2 without performing upsampling to the flow front image.

For training all models, we utilized an Nvidia DGX-1 with 8 Tesla V100 GPUs. This machine is able to train with a batch size of 2048 for both steps of the training process. Especially the Deconv/Conv network is very resource-intensive in terms of its parameters, with the highest consumption of computation power when using 1140 sensor values as input. For this training, the DGX-1 reached its maximum load with batch size 2048, for training runs with smaller sensor grids, the same batch size yielded the best performance compared to larger batch sizes.

## 4 Experimental Evaluation

To put FlowFrontNet to the test, we devised three central evaluation hypotheses. The first is if we can predict a dry spot from simulated pressure sensors at all. The second and central hypothesis is, whether the intermediate step of flow front image generation can improve dry spot classification. This also involves testing the pixel thresholding introduced to obtain sharper internal flow front representations. As a third point that is interesting to evaluate, we investigate the number of sensors that are necessary to classify dry spots sufficiently well, with and without the intermediate flow front generation. We investigate models for 1140, 80, and 20 sensors, to estimate the prediction quality achievable by a reduced number of sensors, see also Table 2.

These particular numbers emerge from taking every 4-th or every 8-th sensor of the full 1140 grid. Taking 1140 sensors corresponds to a sensor distance of 1

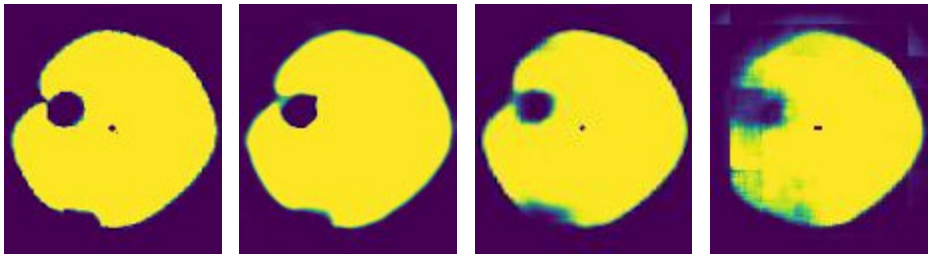
**Table 3.** Accuracy values on three different sensor resolutions for feed-forward baselines and FlowFrontNet

| | Feed-forward | | FlowFrontNet | |
|---|---|---|---|---|
| # Sensors | Threshold | Accuracy | Threshold | Accuracy |
| 1140 | 0.54 | 82.74% | 0.49 | **91.68**% |
| 80 | 0.52 | 79.57% | 0.57 | **83.69**% |
| 20 | 0.49 | 74.68% | 0.51 | **75.22**% |

cm, which is an unrealistically high number. To come closer to reality, we focused on a sensor distance of 4 cm for the 80 sensors because that is within range of physical feasibility and set a baseline with even fewer sensors at 8 cm distance.

### 4.1 Results

*Can machine learning predict dry spots based on sensor inputs?* To start with the first question, we obtained a baseline from the feed-forward network performing the classification task directly from sensor inputs. When optimizing this architecture, we found that larger batch sizes positively affected the evolution of the validation loss, with a sweet spot found at $32{,}768 = 2^{15}$ training instances per batch. The results from experiments suggest the following baselines: the best feed-forward network with two hidden layers achieves an accuracy of 82.73% with 1140 sensors, an accuracy of 79.51% with 80 sensors, and an accuracy of 74.6% with 20 sensors, see Table 3. Unsurprisingly, the more sensors we use, the better the accuracy gets for a network unaware of the underlying fluid dynamics.



(a) Label (b) 1140 sensors (c) 80 sensors (d) 20 sensors

**Fig. 5.** Exemplary flow front predictions in the Deconv/Conv part of FlowFrontNet based on the different sensor counts.

*Does the Deconv/Conv net improve the classification accuracy?* For the second question – the improvement of the classification by generating flow front images using deconvolutional and convolutional layers – the answer is a clear yes, with only a slight limitation. Figure 5 allows us to visually inspect exemplary results from the learned sensor-to-flow-front mapping on the test set. Even for

**Table 4.** Confusion matrices for 1140 and 80 sensors: Feed-forward vs. FlowFrontNet

**For 1140 sensors**

| | | Actual | |
|---|---|---|---|
| | | ¬ Dry spot | Dry spot |
| Pred. | ¬ Dry spot | 40.44% | 8.59% |
| | Dry spot | 8.66% | 42.31% |

| | | Actual | |
|---|---|---|---|
| | | ¬ Dry spot | Dry spot |
| Pred. | ¬ Dry spot | 44.60% | 4.43% |
| | Dry spot | 3.89% | 46.98% |

**For 80 sensors**

| | | Actual | |
|---|---|---|---|
| | | ¬ Dry spot | Dry spot |
| Pred. | ¬ Dry spot | 38.97% | 10.06% |
| | Dry spot | 10.37% | 40.06% |

| | | Actual | |
|---|---|---|---|
| | | ¬ Dry spot | Dry spot |
| Pred. | ¬ Dry spot | 41.60% | 7.44% |
| | Dry spot | 8.87% | 42.10% |

20 sensors, a rough idea of the underlying flow front is obtained, for 80 and more sensors, a fairly accurate image can be reconstructed. Based on that internal representation of the flow front, the accuracy of FlowFrontNet with 1140 sensors as input increases to 91.68%, which is a 9% advantage over the pure Feed-forward Network. For 80 sensors (with pixel-thresholding), the accuracy can be enhanced from 79.51% to 83.69%, a margin of 4%. Furthermore, the 80 sensor FlowFrontNet performs better than the feed-forward network with 1140 sensors which shows that sensor investments could be reduced in favor of encoded simulation knowledge.

Alas, the improvement over the feed-forward network decreases even more with 20 sensors: 75.22%, which is less than 1% of accuracy boost. The spatial information density in a $5 \times 4$ input sensor grid turned out too low to still get a useful representation, which can also be observed in Figure 5d. The image of the flow front is not clear at all and it appears as though there is no dry spot enclosed by resin but rather a jagged flow front. Therefore, we only focus on the models equipped with 80 or 1140 sensor inputs.

To get a more comprehensive performance overview, Table 4 shows the confusion matrices for the 1140 and 80 sensors input, respectively. The models are rather balanced regarding their false positive and false negative shares, but these values are – as always – object to modification by the classification threshold applied to the output probability score. Especially in industrial processes, one type of error can be more favorable than the other, depending on whether false positives (e.g., causing disruptions in the process) or false negatives (e.g., leading to undetected errors in the produced parts) are more acceptable. For the behavior of the classifiers under various classification thresholds, consider the ROC curve in Figure 6. The confusion matrices here are given for the classification threshold value that gives the highest validation accuracy, as per Table 3.

*Is the pixel-thresholding step in FlowFrontNet useful?* By experimenting with possible pixel thresholds (see Section 3) of 0.2, 0.5, and 0.8, we found that 0.8 yielded the best results and is used for the 80-sensor results (also in the already presented results). The accuracy increased from 81.48 to 83.69%. We observed that the training loss decreases steeper and farther without pixel-thresholds but

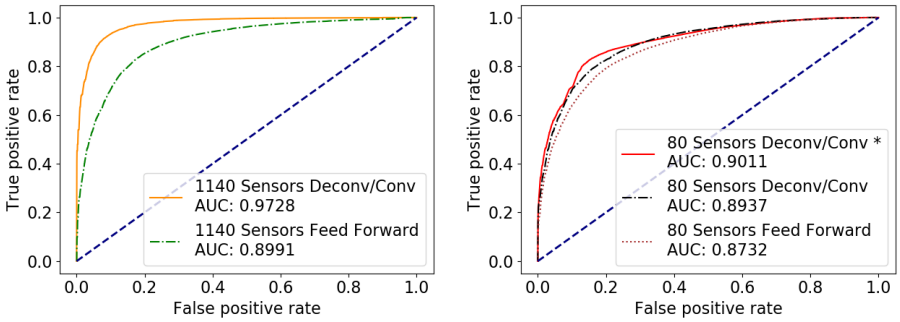



**Fig. 6.** ROC Curves for different models and sensor inputs. * pixel-threshold at .8

the validation loss increases to a greater extent. By contrast, the validation loss for the pixel-threshold model declined steadily – indicating that the thresholding of flow front images regularizes.

Moreover, training more than one epoch without pixel-thresholds produced heavy overfits. Even with an exponential learning rate scheduler and a very low initial learning rate, the training dynamics did not improve. While the validation loss was decreasing, it was at a higher base level than before. With pixel-thresholds in place, training got easier, even without scheduling the learning rate but using a fixed value of $10^{-4}$ with AdamW [10]. Alas, for the smallest 20 sensor grid, pixel-thresholding did not give better results.

The ROC curves and the corresponding area under the curve (AUC) values are shown in Figure 6. The AUC for the .8-pixel-thresholded model is the best of all 80 sensor models, but only by a small margin. The other curves do not hold any surprises, with the 1140 sensor model outperforming all and all other models with similar AUCs. Only the 80 sensor feed-forward net is underperforming.

### 4.2 Discussion - Metrics on Run Level

Our previous evaluations exclusively considered metrics per frame that are each drawn from many independent injection runs. Such a dry spot classification for every frame and, thus, point in time is desirable to closely monitor the RTM process (in a "digital twin"-manner) to intervene at process execution time by adjusting process parameters. Additionally, practitioners care about a judgment concerning the process quality of a single run, e.g., a single produced composite plate. This is similar to lifting single-frame classification to video classifications, e.g., how many dry spot frames make a dry spot run? The first difficulty is to decide when a run counts as a failure and doing so automatically for both the label maps and predictions for all 36,663 runs.

The naive approach would take the last few frames to determine if a run counts as having produced a dry-spot. However, we would miss dry spots that only occur in the middle of a run. These might hint at problems in reality

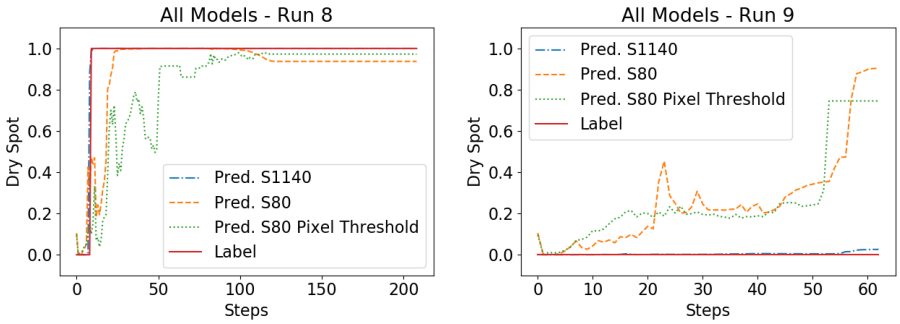



**Fig. 7.** Different classifications of two runs by three different models

and only "close" due to simulation artifacts (e.g., unrealistically increasing the pressure). Another possible criterion counts all dry spot frames and prescribes a minimal amount to mark a run as failed. Alternatively, we could require the dry spot sequences to be contiguous, to avoid listing too many runs as failed if there are single dry spots in the label maps or predictions which could also be artifacts from the label generation process described in Section 2.2. Figure 7 offer some insight into how different run classifiers would behave for two exemplary runs, given the frame-wise ground truth and predictions.

Here, it becomes apparent that the models perform very differently and also confirms that the models taking 80 sensors as input cannot be compared to the 1140 sensor input models. The 1140-sensor-models adhere closely to the label most of the time whereas both 80-sensor-models produce more noise, in different ways. We anticipate further experiments as future work, especially combining the single-frame predictions with another sequence model to classify whole runs – provided that we can obtain or define meaningful run labels.

## 5 Conclusion and Future Work

We presented FlowFrontNet, a deconvolutional/convolutional neural network suitable for detecting dry spots in simulated RTM processes, i.e., improved process monitoring. In doing so, we showed that it is possible to learn the intermediate representation of flow fronts from sensor data by upsampling via a deconvolutional network. This enabled us to reliably classify dry spots on individual frames substantially better than with a feed-forward network using the same sensor input. The classifier makes it possible to intervene during the manufacturing of a single composite plate. We also investigated that the prediction quality decreases with the number of sensors in use, and found that acceptable accuracy requires at least 80 sensors, a sensor grid of 4 cm distance. That is a realistic order of magnitude for real mold sensor layouts whereas 1140 sensors cannot be placed as closely due to cost and wiring issues.

Future work can be divided into short and long term goals that focus on use cases with simulated and actual data, respectively. In the short run, we plan to use the flow front image generated from sensors for other classification or object detection tasks and use sequence models for predicting full runs, with their temporal information. Our long term goal is to use models pre-trained on simulated data for real RTM process data [18], much like sim-to-real applications are being used in reinforcement learning and robotics [13]. In addition to the costly process of actually producing composite plates in a sufficient number for training, reality confronts us with heterogeneous and noisy sensors other than pressure alone. Eventually, we want to generate feedback for process control to avoid rejects. The first step towards this goal, learning the flow front from sensor data and applying it to dry-spot classification, has been successfully achieved.

## Acknowledgments

This research is funded by the Bavarian Ministry of Economic Affairs, Regional Development and Energy in the project CosiMo. We thank Ewald Fauster from Montanuniversität Leoben for his expert advice on the RTM process and Frederic Masseria from ESI for supporting our RTM-simulations.

## References

1. Babb, D.A., Richey, W.F., Clement, K., Peterson, E.R., Kennedy, A.P., Jezic, Z., Bratton, L.D., Lan, E., Perettie, D.J.: Resin transfer molding process for composites (1998)
2. Darcy, H.P.G.: Les Fontaines publiques de la ville de Dijon. V. Dalamont (1856)
3. Grössing, H., Stadlmajer, N., Fauster, E., Fleischmann, M., Schledjewski, R.: Flow front advancement during composite processing: predictions from numerical filling simulation tools in comparison with real-world experiments. Polymer Composites **37**(9), 2782–2793 (9 2016). https://doi.org/10.1002/pc.23474
4. Heuer, H., Schulze, M., Pooch, M., Gäbler, S., Nocke, A., Bardl, G., Cherif, C., Klein, M., Kupke, R., Vetter, R., Lenz, F., Kliem, M., Bülow, C., Goyvaerts, J., Mayer, T., Petrenz, S.: Review on quality assurance along the CFRP value chain - Non-destructive testing of fabrics, preforms and CFRP by HF radio wave techniques. Composites Part B: Engineering **77**, 494–501 (2015). https://doi.org/10.1016/j.compositesb.2015.03.022
5. Heywood Don MacKenzie, J., Bonde Akerlind Parisa Bastani Irene Berry Kandarp Bhatt Alice Chao Eric Chow Valerie Karplus David Keith Michael Khusid Eriko Nishimura Stephen Zoepf, I.: On the Road toward 2050: Report Massachusetts Institute of Technology Potential for Substantial Reductions in Light-Duty Vehicle Energy Use and Greenhouse Gas Emissions. Massachusetts Institute of Technology (2015), http://mitei.mit.edu/publications/
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
7. Lee, D.H., Lee, W.I., Kang, M.K.: Analysis and minimization of void formation during resin transfer molding process. Composites Science and Technology **66**(16), 3281–3289 (2006)

8. Lee, J., Bagheri, B., Kao, H.A.: A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufacturing letters **3**, 18–23 (2015)
9. Liu, B., Bickerton, S., Advani, S.G.: Modelling and simulation of resin transfer moulding (RTM) - Gate control, venting and dry spot prediction. Composites Part A: Applied Science and Manufacturing **27**(2), 135–141 (1996). https://doi.org/10.1016/1359-835X(95)00012-Q
10. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: 7th International Conference on Learning Representations, ICLR 2019 (11 2019)
11. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. vol. 2015 Inter, pp. 1520–1528 (5 2015). https://doi.org/10.1109/ICCV.2015.178
12. Pantelesis, N., Bistekos, E.: Process monitoring and control for the production of CFRP components. SAMPE Conference pp. 5–9 (2012)
13. Peng, X.B., Andrychowicz, M., Zaremba, W., Abbeel, P.: Sim-to-real transfer of robotic control with dynamics randomization. In: 2018 IEEE international conference on robotics and automation (ICRA). pp. 1–8. IEEE (2018)
14. Shelhamer, E., Long, J., Darrell, T.: Fully Convolutional Networks for Semantic Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**, 640–651 (4 2017). https://doi.org/10.1109/TPAMI.2016.2572683
15. Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 2016-Decem, pp. 1874–1883 (9 2016). https://doi.org/10.1109/CVPR.2016.207
16. Shin, H.C., Roth, H.R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., Summers, R.M.: Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. IEEE Transactions on Medical Imaging **35**(5), 1285–1298 (5 2016). https://doi.org/10.1109/TMI.2016.2528162
17. Sorg, C.: Data Mining als Methode zur Industrialisierung und Qualifizierung neuer Fertigungsprozesse für CFK-Bauteile in automobiler Großserienproduktion. Ph.D. thesis, Technische Universität München (2014)
18. Stieber, S.: Transfer Learning for Optimization of Carbon Fiber Reinforced Polymer Production. Organic Computing: Doctoral Dissertation Colloquium 2018 pp. 1–12 (2018)
19. Stieber, S.: FlowFrontNet Checkpoints (2020). https://doi.org/10.6084/m9.figshare.12102714.v1, https://figshare.com/articles/FlowFrontNet_Checkpoints/12102714
20. Stieber, S.: FlowFrontNet Data: Sensor to Flowfront / Dryspot (2020). https://doi.org/10.6084/m9.figshare.12063480.v4, https://figshare.com/articles/FlowFrontNet_Data_Sensor_to_Flowfront_Dryspot/12063480
21. Xu, L., Ren, J.S., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. Advances in Neural Information Processing Systems **2**(January), 1790–1798 (2014)
22. Zhang, J., Pantelelis, N.: Modelling and optimisation control of polymer composite moulding processes using bootstrap aggregated neural network models. 2011 International Conference on Electric Information and Control Engineering, ICEICE 2011 - Proceedings **1**(2), 0–3 (2011). https://doi.org/10.1109/ICEICE.2011.5777841