Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 130 (2024) 145–153

www.elsevier.com/locate/procedia

57th CIRP Conference on Manufacturing Systems 2024 (CMS 2024)

How Domain Knowledge can Improve Machine Learning Surrogates for Manufacturing Process Optimization – a Comparative Study

Bela H. Böhnke[a,*], Aleksandr Eismont[a], Clemens Zimmerling[b], Luise Kärger[b], Klemens Böhm[a]

[a]*Institute for Program Structures and Data Organization (IPD), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany*
[b]*Institute of Vehicle System Technology - Lightweight Engineering (FAST-LB), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany*

* Corresponding author. Tel.: +49-176-477-57012. *E-mail address:* bela.boehnke@kit.edu

Abstract

In various industries, optimizing manufacturing parameters is vital for the efficient production of high-quality products. Traditional methods involve costly production trials and process tuning – particularly when dealing with complex processes and materials such as composites. High-fidelity simulations offer a cost-effective alternative. However, they can be computationally intensive, which often renders them impracticable for iterative optimization. Surrogate model-based optimization (SuMO) provides a solution by using efficient, data-driven approximations. However, existing approaches often overlook valuable domain knowledge, such as material behavior, spatial relationships and optimization objective. We investigate different types of knowledge varying in complexity, difficulty to incorporate and transferability to other domains. In numerical studies on composite manufacturing – specifically, textile draping – we demonstrate that integrating such domain knowledge improves prediction accuracy, reduces optimization iterations, and enhances overall outcomes.

Keywords: Surrogate Modelling; Manufacturing Process Optimization; Domain-informed Machine Learning; Finite-Element-Simulation

1. Introduction and Use Case

Industrial manufacturing processes require careful parametrization for optimal operation in terms of part quality, throughput or efficiency. In current practice, identifying optimal parameters often involves lengthy trial-error campaigns and significant rework for fault correction. High-fidelity process simulations, e.g., based on finite elements (FE), allow to assess manufacturability at the earliest stages of part development [1]. Besides rigorous analysis of process dynamics, their inherently virtual nature also allows for coupling with optimization algorithms, often referred to as *virtual process optimization*. While such a coupling enables automated search for optimal parameters, the computational demands of iterative optimization often make it impractical in real-world applications [2]. One option to reduce the computational burden in virtual process optimization is surrogate model-based optimization (SuMO) [3]. Multiple variants of SuMO exist, which all share the idea of constructing a computationally efficient, data-driven approximation of the expensive simulation – a *surrogate*. This process is referred to as *training* and relies on a priori sampled observations. The resultant surrogate then guides the optimizer in the parameter space.

The more accurate the surrogate, the more efficient the optimization process will be. Since accuracy generally improves with training data, a naïve idea could be to supply more sample simulations. However, the available computational resources usually limit the number of simulations. This in turn makes data-efficiency the key for real-world applicability. In this study, our objective is to enhance surrogate accuracy by leveraging readily available engineering knowledge while maintaining a constant number of simulation samples. We stepwise introduce additional knowledge by domain-agnostic and domain-informed methods and compare the impact on surrogate accuracy. Additionally, we categorize different types of additional knowledge regarding knowledge complexity and assess the difficulty of incorporating the knowledge. Further, we discuss the transferability of our methods to other domains.

1.1. Related Work

Data-driven surrogate models can be used to guide the optimization and identify promising candidate solutions at low computational effort [4]. However, due to their statistical na-

10.1016/j.procir.2024.10.069

ture, they always deviate from the original process, and thus, these candidates may differ from the true optimum of the original process. SuMO tries to sequentially eliminate this bias of the surrogate model by iteratively refining the surrogate model with new observations over the course of optimization [5]. Over the last decades, research on surrogate modeling mostly studied the suitability of different models ranging from simple polynomials and regression trees [6] to stochastic processes [7, 8] and artificial neural networks [9]. Irrespective of the actual model, the studies tend to view surrogates as a phenomenological *input-output*-relation, where adjustable process parameters (*input*), e.g. temperature or pressure, are mapped to a scalar or low-dimensional quality metric (*output*) [3].

However, spurred by advances in machine learning (ML), attempts have been made to process – and also predict – more complex information with data-driven models. For instance, data preprocessing steps like principal component analysis have been introduced to find an information-rich input-space representation [10]. Alternatively, techniques have been studied that do not just output a scalar value but instead predict multi-dimensional quantities, i.e., a full-field estimation of the quality. For applications in material forming, see, e.g., [2, 11, 12].

Overall, the literature shows that the introduction of additional information increases surrogate accuracy. However, most works view this from a methodological perspective, i.e., seek to improve accuracy by algorithmic improvements but tend to disregard other information sources. Such sources can be domain knowledge [13, 14] about material behavior or spatial dependencies, but as of now, no systematic investigations for materials science have been reported.

1.2. Use Case: Textile Forming Optimization

This work considers optimization of the manufacturing process of composites, specifically, continuous-fiber reinforced plastics (CoFRP). CoFRP offer unparalleled weight-specific mechanical properties and are thus increasingly applied across industries . However, their superior properties usually come at a substantial cost: Not only are the materials themselves expensive, also their complex behavior during manufacture entails considerable optimization effort to produce high-quality products. CoFRP-processes generally comprise a process chain with multiple steps [15]. While process parameters need to be optimized for all steps, this work focuses on forming (*draping*) engineering textiles – specifically woven fabrics.

This work revisits the virtual forming optimization problem from [2]. It studies an FE-based forming simulation model of a double-dome geometry, a common benchmark geometry in textile forming. To control the process, 60 spring-guided grippers clamp the textile along its perimeter, as schematically shown in Figure 1. The grippers locally exert restraining forces onto the textile and thereby manipulate its draw-in into the mold. The optimizer can choose gripper spring stiffnesses c_i ($i = 1 \dots 60$) between $0.01 \dots 1.0\,\mathrm{N/mm}$.

Due to their textile architecture, woven fabrics have a low shear stiffness compared to their tensile stiffness in warp and weft direction. This makes in-plane shear the dominant deformation mechanism. One can quantify it by the shear angle γ, as visualized in Figure 2.

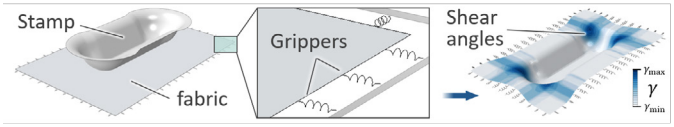

Fig. 1. Left: Forming simulation setup with 60 grippers along the textile perimeter, visualized by springs. Right: Example shear angle distribution after forming. Some springs stretch and locally introduce a restraining force. [2, 16]

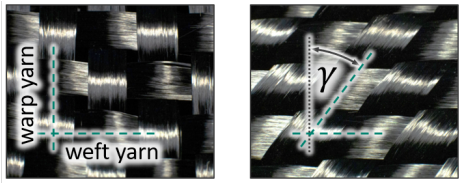

Fig. 2. Shear deformation of a woven fabric measured by the shear angle γ. [17]

However, fabrics cannot undergo arbitrarily large shear deformations but show a forming limit, the *locking angle* γ_{lock} [18]. Shearing beyond γ_{lock} increases the likelihood of defects like wrinkling and textile folding. Also, high shear angles impede resin infiltration and may lead to uninfiltrated regions (*dry spots*) which can compromise the structural performance [16, 19]. Therefore, the shear angle is a crucial quality indicator during the forming process, and it is typically minimized by finding the optimal gripper spring stiffness combination.

2. Considered Domain Knowledge and Inclusion Methodes

We investigate the effect of domain knowledge on surrogate accuracy. As Table 1 summarizes, we study three different approaches to include domain knowledge. We selected our approaches to cover typical levels of complexity regarding: (1) Contained domain knowledge, (2) required effort to include the knowledge, and (3) transferability to other manufacturing processes. Our approaches are outlined in the following.

Table 1. Investigated domain knowledge

Example	Knowledge	Inclusion	Transferability
Geometry-strain relation	simple	simple	general
Gripper-tensile-force relation	complex	complex	specific
Objective Alignment	complex	simple	general

2.1. Geometry-Strain Relation

In manufacturing and general engineering, we can typically expect a complex relation between component geometry, manufacturability and structural performance.

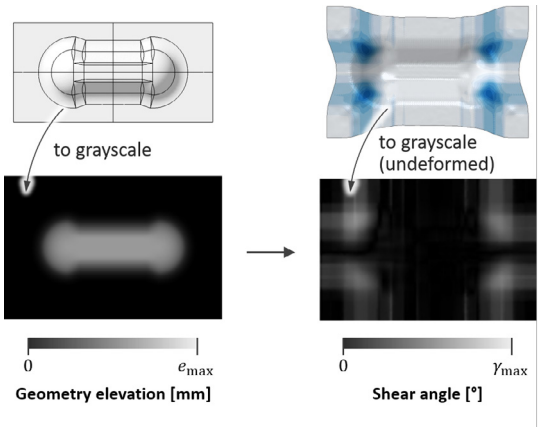

Fig. 3. Encoding the tool geometry (top left) and material shear (top right) as grayscale images (bottom).

Knowledge. One such relation stems from the deformation mechanism of woven fabrics: Shear deformation will mainly form in doubly-curved geometry regions. That is, a close spatial relation between geometry and high shear angles can be expected. We deem this rather broad and qualitative statement a comparably simple form of domain knowledge. It 'only' requires a description of the geometry and the part quality distribution, here the set $\boldsymbol{\gamma}$ of shear angles.

Inclusion. For textile forming, this has proven comparably straightforward: As proposed in [20, 21] and later confirmed in [22], images are well-suited to describe such spatial relations in textile forming: One can encode the tool geometry and material shear as a grayscale image using the local elevation from the tool separation plane and, likewise, the local shear angles as shown in Figure 3. At the same time, images are suitable data formats for ML techniques, which makes incorporation straightforward.

Transferability. We hypothesize that a spatial-aware surrogate model achieves better generalization performance than its classical *input-output* counterpart. This is because the spatial-aware model requires less training data to achieve the same accuracy. Such qualitative geometry-process-relations are widespread in manufacturing and general engineering: Consider, for instance, fiber reorientation along the flow paths in molding processes or stress concentrations at geometrical notches. Thus, we expect good transferability to other domains.

2.2. Gripper-Tensile-Force Relation

We further utilize domain knowledge to encode the positions and the area of influence of the grippers. The grippers actuate the textile locally, and thus, we again expect a close spatial relation between grippers and the textile response.

Knowledge. A major and a minor mechanism is assumed to govern the effect of grippers on the fabric: (1) The continuous fibers can transmit large tensile loads along their axis, and thus, each gripper will actuate the fabric yarns connected to its point of attack in a line-wise manner. (2) Shear forces are transmitted to neighboring yarns via friction at the warp-weft-crosspoints, although on a much lower scale. However, as the frictional forces at the crosspoints add up along the fiber axes, more and more neighboring yarns are actuated. As a result, the grippers' area of influence will spread to a certain extent along its line of action. We deem these material-specific mechanisms a complex form of knowledge.

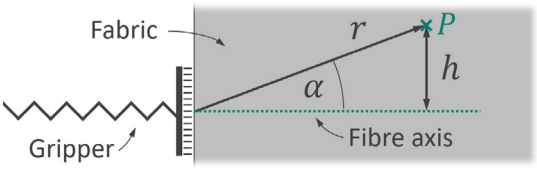

Fig. 4. Relative rotation angle of an affected material point P and fiber axis.

Inclusion. We resort again to image representations and consider two possible approaches: a *stiffness*-encoding and, extending it, a *force*-encoding. Both encodings seek to approximately represent the two material mechanisms. Since the nonlinear, multi-scale behavior of woven fabrics defies a rigorous mechanics-based, closed-form model of the gripper effects, we lean on simplified elasticity theory for orthotropic materials. Please note that the following is not a rigorous derivation based on continuum mechanics but a pragmatic adaptation to comply with engineering understanding. We assume that the gripper influence behaves roughly similarly to the stiffness of a unidirectional fiber under rotation. That is, it is maximal along the fiber axis (reduced plane-stress stiffness Q_{11}) but diminishes with relative rotation α to the fiber axis (Figure 4). See, e.g., [23] for the transformation equations of the reduced stiffnesses Q_{ij} under rotation.

In our study, we are more focused on relative values than absolute ones. Therefore, we normalize the stiffnesses relative to the stiffness maximum Q_{11} between 0 and 1 via $q_{\mathrm{rel}} = k\, Q_{ij}/Q_{11}$. This normalization process employs an attenuation factor k, which reduces the signal's intensity exponentially in further distance h from the fiber direction and reflects the lower friction forces. Specifically, we set $k = \exp\left(-(h/l_{\mathrm{aff}})^2\right)$. l_{aff} represents a length which increases in proportion to the distance r: $l_{\mathrm{aff}} = [1 - \exp(-r/r_{\mathrm{max}})] \cdot [l_{\mathrm{aff}\,\infty} - l_{\mathrm{aff}\,0}] + l_{\mathrm{aff}\,0}$ with $l_{\mathrm{aff}\,\infty} = 30\,\mathrm{mm}$, $l_{\mathrm{aff}\,0} = 10\,\mathrm{mm}$ and $r_{\mathrm{max}} = 200\,\mathrm{mm}$. The relative stiffness q_{rel} models the grippers' areas of influence while the spring stiffnesses c scale them up and down so that we obtain the *stiffness*-encoding I_{C} via $I_{\mathrm{C}} = c \cdot q_{\mathrm{rel}}$.

We give two encoding examples for I_{C} in Figure 5 (center top), one with a high stiffness (dark) and one with a lower stiffness (bright). Clearly, the distributions reproduce the engineering understanding: From the grippers' points of attack (blue and yellow markers) I_{C} is maximal in warp or weft direction, respectively, and gradually widens in perpendicular direction.

To obtain the *force*-encoding, we extend the stiffness encoding. Consider the situation shown on the right of Figure 5. Spring A (blue marker) is barely stretched, while spring B (yellow marker) experiences considerable stretch, i.e., $u_{\mathrm{B}} >> u_{\mathrm{A}}$.

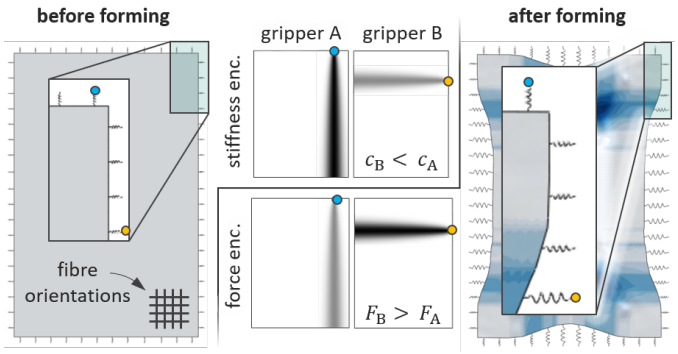

Fig. 5. Gripper stiffness and force impact distribution I_C and I_F. Gripper forces F_i computed from stiffness c_i and reference-displacement u_{ref} by $F_i = c_i \cdot u_{ref}$.

Since the exerted force F of a gripper depends not only on its spring stiffness c but also on its stretch u ($F = c \cdot u$), the stiffness of each griper contributes to the forming process to different extents. Thus, we obtain the force-encoding $I_F = I_C \cdot u_{ref}$ by multiplying the gripper stiffness I_C with a reference-displacement u_{ref} to factor in the expected spring stretch during forming. Here, u_{ref} comes from an additional forming simulation, where all gripper springs have a uniform stiffness of $c = 0.5\text{N}\,\text{mm}^{-1}$. By comparing the encodings of I_F (cf. Figure 5 center bottom) and I_C (cf. Figure 5 center top) it becomes apparent that the force-encoding of the gripper-tensile-force relation is different depending on how much the fabric is locally drawn into the mold: Although spring B has a lower stiffness than spring A ($c_A > c_B$), it stretches much more ($u_A << u_B$) and thus exerts a higher force during forming. Consequently, in the force-based encoding, spring B is more pronounced.

As with the geometry-strain relation, including the image-based encoding into the surrogate is easy. However, we deem the incorporation approaches complex because of the mechanical understanding required to obtain the encoding.

Transferability. These image-based encodings are highly process-specific, because of the material-specific deformation mechanisms. Thus, they may not be directly applicable to other processes, although it is certainly conceivable to devise different representations for other processes.

2.3. Alignment of training and optimization objective

Optimization amounts to minimization of an scalar objective function value. Historically, surrogate training aims at predicting this scalar objective function value accurately. However, an accurate prediction of the objective function is important only near minima. Thus, predictions close to minima are more important and require highest-possible accuracy. Classical surrogate training does not reflect this difference in importance, though, but weighs all data equally. Recent work has shown that full-field predictions are beneficial for accuracy [2, 12] because it allows learning relations between neighboring regions. However, full-field predictions instead of scalar objective functions even compound this difference in importance. This is, because in a full-field many elements contribute only little to the objective function, making their accurate prediction less important. However, current work weighs them equally to the – often few – elements that contribute considerably to the objective function. We name this issue *training-objective-bias*.

Knowledge. The objective function already contains well-formalized domain knowledge about what constitutes part quality and possibly defect allowables. Specifically, the objective function quantifies the importance of different elements with respect to the overall part quality. For the n-th gripper configuration, the shear strain is pixel-wise encoded in an image $\boldsymbol{\gamma}_n = (\gamma_{n1}, \ldots, \gamma_{nP})$ with P being the pixel count. In accord with [11], we assume the norm $o(\boldsymbol{\gamma}_n) = \|\boldsymbol{\gamma}_n\|_k = (\sum |\gamma_{np}|^k)^{1/k}$ with $k = 4$ as the objective which balances suppression of maximum shear and general shear formation. By incorporating this – often complex – knowledge into the surrogate, we expect to identify important regions and reflect their importance during training.

Inclusion. We propose a novel method – *Objective Alignment* (OA) – that makes use of this knowledge during model training. OA seeks to align the training objective with the optimization objective, thereby counteracting the training-objective-bias. During training, the pixel-wise loss of the shear strain field (Figure 3, right) is weighted by its pixel-wise importance with respect to the objective function. We argue that the importance of a ground truth value γ_{np} of image n and pixel p is quantified by the influence W_{np} the value γ_{np} has on the objective function $o(\boldsymbol{\gamma}_n)$ calculated over the whole ground truth strain field $\boldsymbol{\gamma}_n$. We quantify this influence W_{np} via backpropagation as the gradient and then normalize the overall importance matrix W_n with a min-max normalization to the interval $[0.1, 1]$ to obtain pixel weights w_{np}:

$$W_{np} = \frac{\delta o(\boldsymbol{\gamma}_n)}{\delta \gamma_{np}}, \quad w_{np} = \left|_{0.1}^{1} W_{np}\right|_{\min(W_n)}^{\max(W_n)} \tag{1}$$

. Note that we do not normalize to 0 so that any pixel has at least some influence to avoid random predictions $\hat{\gamma}_{np}$. The obtained weights then quantify the contribution of each pixel to the overall objective-aligned loss. The Mean Absolute Error (MAE) (per image n) with OA correction reads:

$$\text{MAE}_{\text{OA},n} = \sum_{p=1}^{P} w_{np} \left|\gamma_{np} - \hat{\gamma}_{np}\right| \tag{2}$$

and analogously for other losses such as Mean Square Error (MSE), see Appendix A.

Overall, we expect OA to reduce the necessary amount of calibration data, i.e., surrogate refinement iterations, and allow for faster identification of optimal parameters. OA is applicable to any differentiable objective function without further engineering effort. Thus, we deem OA a simple incorporation method.

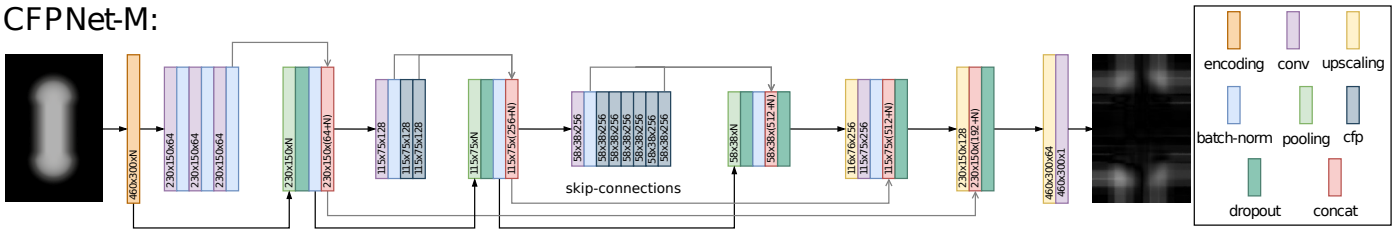

Fig. 6. The CFPNet architecture, which is the surrogate model of our choice because of its better overall performance compared to all other investigated architectures.

Transferability. Unlike our domain encoding (stiffness-/force-encoding), the inherent domain knowledge about the optimization objective is readily available in any automated parameter optimization context such as SuMO, and already formalized within the objective function. Thus, OA ought to be excellently applicable to objective functions from other domains. This makes it generally useful across disciplines.

3. Numerical Studies and Results

Having outlined the envisaged domain knowledge and suitable domain-based inclusion methods, this section presents the setup of our numerical studies and the observed effects on surrogate accuracy and performance for SuMO. First, we introduce the common study setup. We then discuss various surrogate architectures and existing domain-independent knowledge inclusion methods, which we use as baselines in our studies.

3.1. General Study Setup

We perform studies for all three types of domain knowledge similarly: We scale the data to an [0, 1] interval with respect to the physically possible shear minimum (0°) and maximum values (90°). We train each investigated surrogate model with the Adam optimizer [24] with an initial learning rate of 0.001 and a batch size of 8. We perform each study on training sets of sizes between 100 and 900 to investigate how well the surrogate can generalize from different amounts of data. The generalization capability is especially important for SuMO, where each data point is costly, which is why our discussion of results will focus on smaller training set sizes up to 500. For each training set size, we perform a 5-fold cross-validation with a separate test set of constant size 100. For each validation we take the end results after early stopping with a 60-epoch patience period or after a maximum of 300 epochs. We then report their mean and 95 % confidence interval per training set size.

3.2. Geometry-Strain Relationship

The first kind of additional information we include in the surrogate model is the *geometry-strain-relation*, cf. Section 2.1. There we represent the geometry and the shear strain field by grayscale-images (Figure 3). This provides root-cause information (doubly-curved geometry regions) towards the formation of the shear field, which the grippers then manipulate. We accordingly select image-processing architectures for the surrogate, namely convolutional neural networks (CNN).

3.2.1. Surrogate Architectures

We use the Multi-layer perceptron (MLP) from the original paper [2] as our baseline. It takes the 60 spring stiffnesses c_i, $i = 1 \dots 60$ as input and estimates the full strain field. It neither has knowledge of spatial relationships or stamp geometry nor does it feature convolutional layers.

We compare this baseline (MLP) to three CNN architectures for image-to-image tasks: A classical encoder-decoder-architecture from [17, 20] for geometry-to-strain prediction, a *U-Net* architecture [25], and the state-of-the-art *CFPNet-M* architecture [26] (cf. Figure 6). Note that the U-Net and the CFPNet use skip connections, which allow information to flow from previous layers to subsequent layers to prevent the problem of vanishing gradients in deep networks. The CFPNet additionally introduces the new *Channel-wise Feature Pyramid* (CFP) -modules, which facilitate learning of features of varying sizes. A regular multi-path architecture includes the non-image spring stiffnesses into the CNNs. As Figure 7 shows, they are fed through a separate network which is concatenated to the bottle net stage of the CNNs. Specifically, we reuse the MLP from [2] for the non-image path.

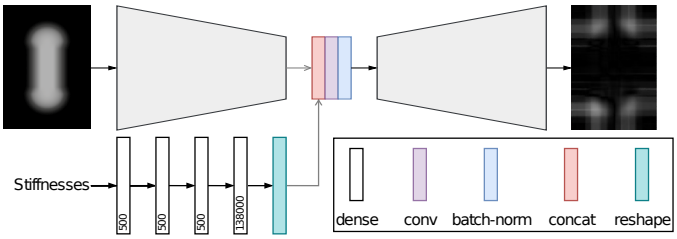

Fig. 7. The multi-path architecture for merging non-image data with image data.

We train all models with the MSE-loss and evaluate them after training in terms of MAE and RMSE_{obj}. See Appendix A for their definition. The RMSE_{obj} considers the objective function and, thereby, is especially informative regarding optimization performance.

3.2.2. Results

Figure 8 visualizes the results. Clearly, U-Net and CFPNet outperform all other architectures when training on small data, i.e., ≤ 250 samples. For 250 data points, the MAE decreases by ≈ 65% and the RMSE_{obj} even by 75% of the MLP (baseline) with practically negligible scatter. The U-Net performs exceptionally well and almost reaches its maximum performance with only 100 data points. Interestingly, the MLP is able to catch up from 500 data points onwards and even out-

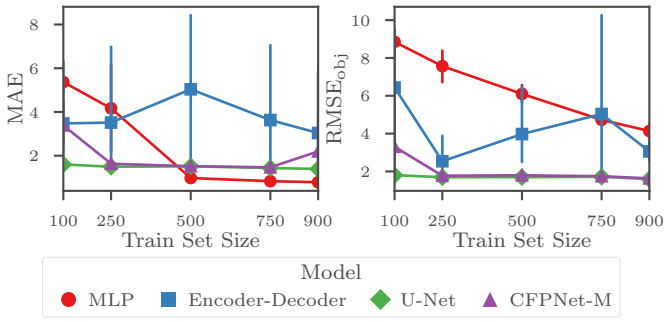

Fig. 8. (left) model comparison of MAE between predicted strain fields; (right) model comparison in RMSE based on the objective function. Small values imply better performance.

performs the CNNs regarding the MAE. However, we expect the performance on $\mathrm{RMSE}_{\mathrm{obj}}$ to be more informative regarding optimization performance, where the CNNs consistently keep their superiority. The encoder-decoder performance proves entirely unstable – presumably due to missing skip connections – and we exclude it from further investigations. Nonetheless, we can observe that every architecture that makes use of domain knowledge performs significantly better for small training set sizes than the MLP without domain knowledge.

3.3. Encoding of Grippers

We further study the effect of different gripper representations on surrogate accuracy. Specifically, we want to assess the suitability of our domain-informed gripper-tensile-force encodings I_{C} and I_{F} (Figure 5), respectively, compared to domain-independent encodings.

3.3.1. Gripper Encoding Methods Investigated

We study two CNN architectures, U-Net and CFPNet. Our baseline uses a vector-valued gripper encoding, which is fed into the CNNs via the multi-path architecture, cf. Figure 7. We compare these baselines to models with 'image-only' gripper encoding. Specifically, we use two domain-agnostic encodings and our domain-informed encoding I_{C} and I_{F}. As domain-agnostic methods we use *naïve copy* and *DeepInsight+* to transform the vector-data to image-data without using domain knowledge. All tested models now make use of the geometry-strain-relation from Section 2.1.

Naïve copy populates for each value of a vector a matrix of the desired image input size. In our case, this amounts to 60 matrices for the 60 gripper stiffnesses and to $2 \cdot 60 = 120$ matrices for their positions, i.e., an input shape $H \times W \times 3 \cdot 60$, with image height H and width W. While simple and easy to implement, this method can lead to excessive memory consumption, as it replicates values for each vector element.

DeepInsight+ is a combination of two state-of-the-art domain-independent techniques to transform non-image data into image data. The first technique is based on the finding that combining multiple data representations generally benefits learning. While the original work [27] proposes to combine three simple encoding schemes – *Row-Wise Copy*, *Distance Matrix*, and *Equidistant Bars* – we add the DeepInsight encoding taken from [28]. Notwithstanding algorithmic details, DeepInsight utilizes the property that CNNs compute neighboring pixels together and that such pixels share information. The DeepInsight methodology automatically places similar features close together while distancing dissimilar ones. The placement of features involves a dimensionality reduction on the whole training set and projects high-dimensional feature vectors onto a two-dimensional image plane. The feature values of a specific input vector are then mapped to these locations producing one specific image for one specific vector.

Note that DeepInsight requires a fixed training data set. Sequential data acquisition schemes such as SuMO require recalculating the DeepInsight mapping and retraining the entire network at each iteration, increasing the computational effort.

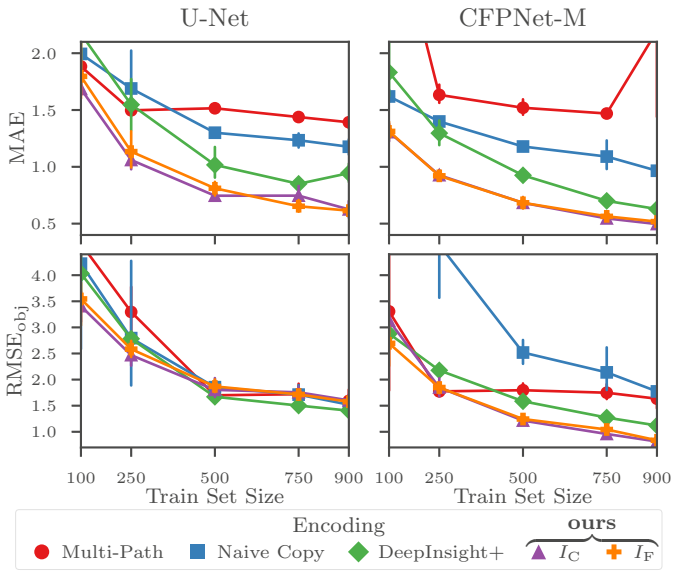

Fig. 9. Comparison of gripper-encoding approaches for CFP-Net regarding MAE and $\mathrm{RMSE}_{\mathrm{obj}}$.

3.3.2. Results

Figure 9 shows the results obtained with U-Net and CFP-Net. Our domain-driven encodings I_{C} and I_{F} almost consistently outperform the domain-independent encoding schemes (cf. Section 3.3.1), especially for small training sets. While increasing the training set size benefits all encodings, the multi-path architecture with vector-valued gripper encoding exhibits performance plateaus between 250 and 500 data points. Interestingly, there is practically no difference between gripper stiffness and gripper-force encoding. This might imply that the spatial distribution of the gripper influence is more important than the signal intensity. For the optimization-relevant sparse-data situations (100 samples), our domain encoding performs best on the CFPNet-architecture and reduces the MAE by $\approx$ 120% and the $\mathrm{RMSE}_{\mathrm{obj}}$ by $\approx$ 20%. We observe inferior performance of U-Net throughout our studies. Thus, we select CFPNet as the model of our choice and for discussion in the following.

3.4. Alignment of Training and Optimization Objective

The last kind of additional information we study is objective alignment (OA), cf. Equation (2). The objective function of an engineering problem contains complex and well-formalized domain knowledge, which OA seeks to leverage. Although this knowledge is case-specific, it is always available in any automated parameter optimization context across disciplines. However, a key challenge remains: developing a method to effectively exploit this information in the surrogate model.

3.4.1. Investigated Loss Functions

To evaluate the suitability of OA, we compare the resulting surrogate accuracy obtained by OA to the accuracy obtained with other loss functions. Specifically, we compare our MAE_{OA} (Equation (2)) to the commonly used MAE, MSE, and also the *structural similarity* (SSIM) loss [29] and a *domain-regularized* loss (DRL) [30].

The SSIM-loss considers changes in *structural information*, *luminance*, and *contrast* of an image. Similar to CNNs, SSIM operates on a number M of image windows rather than individual pixels. SSIM reads per image n:

$$\text{SSIM}_n = 1 - \frac{1}{M}\sum_{m=1}^{M}\frac{(2\mu_{\boldsymbol{\gamma}_{nm}}\mu_{\hat{\boldsymbol{\gamma}}_{nm}} + c_1)(2\sigma_{\boldsymbol{\gamma}_{nm},\hat{\boldsymbol{\gamma}}_{nm}} + c_2)}{(\mu^2_{\boldsymbol{\gamma}_{nm}} + \mu^2_{\hat{\boldsymbol{\gamma}}_{nm}} + c_1)(\sigma^2_{\boldsymbol{\gamma}_{nm}} + \sigma^2_{\hat{\boldsymbol{\gamma}}_{nm}} + c_2)} \quad (3)$$

where $\boldsymbol{\gamma}_{nm}, \hat{\boldsymbol{\gamma}}_{nm}$ are windows in the original $\boldsymbol{\gamma}_n$ and the predicted $\hat{\boldsymbol{\gamma}}_n$ image, and μ_* and σ_* is the mean and the variance of such a window, while $\sigma_{*,*}$ is the covariance between windows. The constants c_1, c_2 are added to avoid instability. We use the standard hyperparameters, except for the window size, which we set to 5; for more details on SSIM and its parameters, see [29].

The DRL adjusts the loss function with penalty terms to reflect domain constraints and is used to include domain knowledge in several works [31–33]. It consists of two parts: a *label-based* part for the training data and a *knowledge-based* part for the domain knowledge. During model training, the optimizer tries to satisfy both parts simultaneously using a tradeoff hyperparameter α. We use the difference of weights $w_{np}, \hat{w}_{np}$ for pixel p in image n, based on the predicted shear strain $\hat{\gamma}_{np}$ and the ground truth shear strain γ_{np} as in Equation (1) as knowledge-based part:

$$\text{MAE}_{\text{DRL},n} = \underbrace{(1-\alpha)\sum_{p=1}^{P}\left|\gamma_{np} - \hat{\gamma}_{np}\right|}_{\text{Label-based}} + \underbrace{\alpha\sum_{p=1}^{P}\left|w_{np} - \hat{w}_{np}\right|}_{\text{Knowledge-based}}. \quad (4)$$

The intuition behind this formulation is to punish pixels that should be important for the objective but whose predictions are not, and vice versa. DRL can be combined with any standard loss function. We did this in our evaluation with MAE, MSE, and SSIM. To select the α hyperparameter, we conducted a grid search in the range $[0, 1]$ with a step size of 0.1 with a fixed training set size of 500. The optimal value for α was determined to be 0.5 and used for all subsequent numerical studies.

Table 2. Evaluation metrics for different loss functions and train sizes. Bold entries mark the best result within one metric

Samples	Loss	Evaluation Metrics RMSE	MAE	RMSE_{obj}
100	MAE	**1.37 ± 0.05**	**1.01 ± 0.03**	3.00±0.46
	MSE	1.74 ± 0.12	1.31 ± 0.10	3.13±0.33
	SSIM	1.74 ± 0.23	1.03 ± 0.03	15.78±7.55
	DRL MAE	1.94 ± 0.40	1.41 ± 0.29	3.43±0.67
	DRL MSE	1.91 ± 0.05	1.42 ± 0.04	3.06±0.27
	DRL SSIM	1.68 ± 0.11	1.24 ± 0.09	5.26±1.58
	OA MAE	1.63 ± 0.10	1.18 ± 0.08	**2.36 ± 0.20**
	OA MSE	1.77 ± 0.08	1.34 ± 0.06	2.44±0.25
250	MAE	**1.02 ± 0.04**	**0.74 ± 0.03**	1.66±0.18
	MSE	1.25 ± 0.09	0.93 ± 0.06	1.85±0.23
	SSIM	1.03 ± 0.08	0.75 ± 0.04	4.83±4.73
	DRL MAE	1.28 ± 0.22	0.93 ± 0.14	1.75±0.18
	DRL MSE	1.42 ± 0.10	1.05 ± 0.06	1.92±0.20
	DRL SSIM	1.26 ± 0.19	0.93 ± 0.13	2.55±0.23
	OA MAE	1.17 ± 0.07	0.85 ± 0.05	**1.27 ± 0.06**
	OA MSE	1.33 ± 0.06	1.00 ± 0.07	1.62±0.16
500	MAE	**0.75 ± 0.03**	**0.55 ± 0.02**	1.28±0.10
	MSE	0.91 ± 0.06	0.68 ± 0.05	1.21±0.09
	SSIM	0.79 ± 0.07	0.56 ± 0.02	5.36±5.74
	DRL MAE	0.83 ± 0.03	0.60 ± 0.03	1.23±0.10
	DRL MSE	0.97 ± 0.06	0.72 ± 0.04	1.39±0.07
	DRL SSIM	0.86 ± 0.07	0.64 ± 0.05	1.92±0.15
	OA MAE	0.87 ± 0.03	0.62 ± 0.03	**0.88 ± 0.11**
	OA MSE	1.02 ± 0.10	0.75 ± 0.08	1.10±0.09

3.4.2. Results

Table 2 summarizes the average accuracy and the 95%-confidence interval for models trained with a given loss and evaluated on different evaluation metrics. For brevity, Table 2 concentrates on sample sizes from 100 to 500 as they are most relevant for data-efficient SuMO. The following observations were consistent across all sample sizes, though. If the loss function is similar to the evaluation metric, i.e., training and evaluation objectives are aligned, performance is best. This supports our hypothesis that objective alignment is important. To estimate product quality, we deem the RMSE_{obj} metric most relevant. For RMSE_{obj} we see that MAE_{OA} and MSE_{OA} outperforms every other loss function. It is noteworthy that for all evaluation metrics, some version of the MAE performs best. We conclude that MAE is the most suitable loss for general strain field prediction and MAE_{OA} for product quality prediction.

4. SuMO with Domain Knowledge

After evaluating the surrogate performance for each knowledge type separately, we assess their suitability for SuMO with all knowledge included. We evaluate four distinct optimization strategies, a non-surrogate approach, and three different SuMO strategies: I) This approach is a classical evolutionary algorithm (EA) without surrogate as in [11]. II) The second

Table 3. Comparison of final optimization results for different metrics. Bold values mark the best results.

Surrogate	Start Results Quality $\bar{o}(\hat{y})$	Start Results γ_{max} in °	End Results Quality $\bar{o}(\hat{y})$	End Results γ_{max} in °	AUC
EA	385.24	47.29	380.18	43.73	342473
MLP	382.61	43.92	379.29	43.70	341716
MLP - MC	382.61	43.92	377.70	42.72	341251
CFPNet-M-MC	382.61	43.92	**373.77**	**41.03**	**339381**

approach (MLP) aligns with [2, 11]. It uses the MLP model from [2] as a surrogate and employs an EA to minimize the objective function while iteratively refining the surrogate. III) The third approach (MLP-MC) extends the MLP model by integrating *MC-Dropout*[1] for uncertainty estimation and Bayesian Optimization with Expected Improvement. No domain knowledge is involved so far. IV) Finally, our approach (CFPNet-M-MC) substitutes the MLP-MC surrogate with the CFPNet-M and includes all domain knowledge. CFPNet-M-MC also uses MC–Dropout to enable Bayesian Optimization.

4.1. Numerical Study Setup

Since the optimization approaches are stochastic, performance evaluations on a single optimization run are not meaningful. Thus, we reevaluate each optimization strategy three times. To validate our domain-informed SuMO approach, we need to run the entire SuMO procedure assuming that we have no initial data. This requires a ground truth simulator. Since a single FE forming simulation takes up to 2 hours, repeated optimization runs with hundreds of simulations are not feasible. Hence, we replace the FE simulator with a more efficient oracle model based on our best-performing model: CFPNet, with all domain knowledge, trained with MAE loss on the entirety of the available dataset of 900 simulations. In our study, we treat data from the oracle as ground truth data. Since we only have to train this model once, we use a model with higher learning capacity, i.e., it has 128 convolutional channels [26].

We initiate all SuMO strategies with a design of 100 simulations generated via a Sobol sequence. The EA-method, i.e. strategy I), starts without simulations. We restrict each method's access to the oracle to a total of 1000 simulations, which includes the initial 100 simulations.

4.2. Results

Table 3 compares the start and end results of the optimization. Our CFPNet-M-MC method outperforms every other method in every metric. Compared to the start shear angle of EA, CFPNet-M-MC improves on the maximum shear angle γ_{max} by 6.3° or 13.24 %, respectively. We further evaluate the area under the curve (AUC, see Appendix A). Besides the optimization result, it factors in how fast the optimization converges. Again, our method outperforms every other method, indicating that it performs better almost everywhere during the optimization.

[1] Monte Carlo Dropout randomly switches off a certain number of neurons during model training and evaluation to calculate estimation uncertainty.

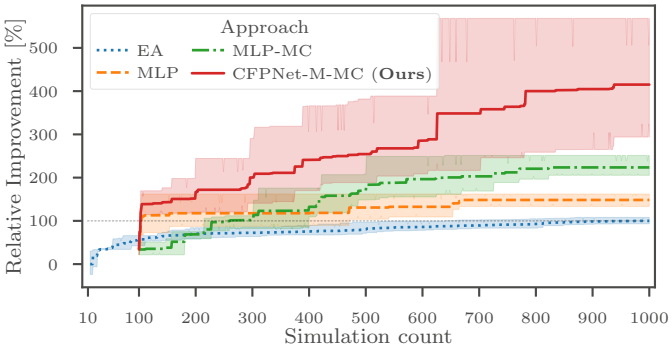

Fig. 10. Improvement during optimization with 95% confidence interval.

Figure 10 shows optimization convergence in terms of *Relative Improvement* (RI) in iteration i according to: $\mathrm{RI}_i = (\bar{o}(\boldsymbol{\gamma})_i - \bar{o}(\boldsymbol{\gamma})_{\mathrm{EA,s}})/(\bar{o}(\boldsymbol{\gamma})_{\mathrm{EA,e}} - \bar{o}(\boldsymbol{\gamma})_{\mathrm{EA,s}})$, where $\bar{o}(\boldsymbol{\gamma})_{\mathrm{EA,s}}$ and $\bar{o}(\boldsymbol{\gamma})_{\mathrm{EA,e}}$ is the part quality of the EA at optimization start and end, respectively, averaged for three runs. Likewise, $\bar{o}(\boldsymbol{\gamma})_i$ is the part quality of the corresponding optimization strategy. In loose terms, RI quantifies how much total optimization potential of the EA-performance another strategy has exhausted in iteration i. The graphs in Figure 10 confirm the AUC result: In every iteration, our approach finds a process configuration that is, on average, better than any of its competitors. While the results of our method scatter more than the baselines, they scatter in the direction of even better results. We assume the increased scatter stems from network initialization effects which may be larger for CNNs than for MLPs.

5. Conclusions

We investigated different methods of including domain knowledge and proposed objective alignment (OA), a new method that is generally useful across domains for including complex domain knowledge in surrogate models for SuMO. All investigated methods in isolation significantly improve the surrogate model. For the combination, we have shown that domain knowledge in SuMO outperforms every state-of-the-art model by a significant margin.

Further research is envisaged: In particular, we want to apply our methods to other domains to see if we can reach similar surrogate improvements. Domain-independent methods like OA are directly applicable and promise to improve the surrogate in any domain. Domain-specific types of knowledge like geometry-process-relations may require adaptations of the incorporation methods and different network structures for representation, e.g., graph neural networks.

Acknowledgement

This work was funded by the German Research Foundation (DFG) as part of projects 452183896 and 459291153. In

terms of the forming use case, this work was initiated in the IGF project OptiFeed (21949 N), funded by the AiF (BMWK), and continued in Subprojects T2 of the DFG AI Research Unit 5339.

Appendix A. Loss Functions for Neural Networks

Mean Absolute Error (MAE) and (Root) Mean Squared Error (MSE, RMSE) are commonly used as loss functions and evaluation metrics to assess the quality of a trained network. This work further uses the RMSE on the objective function ($\mathrm{RMSE}_{\mathrm{obj}}$) and the *Area Under the Curve* (AUC), defined as:

$$\mathrm{MAE}_n = \sum_{p=1}^{P} \left|\gamma_{np} - \hat{\gamma}_{np}\right|, \qquad \mathrm{RMSE}^2_{\mathrm{obj},n} = \left(o(\boldsymbol{\gamma}_n) - o(\hat{\boldsymbol{\gamma}}_n)\right)^2 \tag{A.1, A.3}$$

$$\mathrm{MSE}_n = \sum_{p=1}^{P} \left(\gamma_{np} - \hat{\gamma}_{np}\right)^2, \quad \mathrm{AUC} = \frac{1}{2}\sum_{i=1}^{I-1}\left(o(\hat{\boldsymbol{\gamma}}^*_{i+1}) + o(\hat{\boldsymbol{\gamma}}^*_{i+1})\right) \tag{A.2, A.4}$$

Therein, P is the number of pixels in an image n, with γ_{np} denoting the true and $\hat{\gamma}_{np}$ the predicted value of the p-th pixel. Further, I is the number of optimization steps, and $\hat{y}^*_i$ is the strain field of the best product found up to the current iteration i. Averaging the per-image loss across all N images gives the total loss. In addition, $\mathrm{RMSE}_{\mathrm{obj}} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\mathrm{RMSE}^2_{\mathrm{obj},n}}$ requires taking the square root.

References

[1] Mourtzis, D.. Simulation in the design and operation of manufacturing systems: state of the art and new trends. Int J Prod Res 2020;58:1927–1949.

[2] Zimmerling, C., Schindler, P., Seuffert, J., Kärger, L.. Deep neural networks as surrogate models for time-efficient manufacturing process optimisation. ESAFORM 2021;MS11:3882.

[3] Koziel, S., Leifsson, L.. Surrogate-based modeling and optimization. Springer; 2013.

[4] Gorissen, D., Couckuyt, I., Demeester, P., Dhaene, T., Crombecq, K.. A surrogate modeling and adaptive sampling toolbox for computer based design. JMLR 2010;11:2051–2055.

[5] Bouhlel, M.A., Hwang, J.T., Bartoli, N., Lafage, R., Morlier, J., Martins, J.R.. A python surrogate modeling framework with derivatives. Adv Eng Softw 2019;135:102662.

[6] Smarra, F., Jain, A., de Rubeis, T., Ambrosini, D., D'Innocenzo, A., Mangharam, R.. Data-driven model predictive control using random forests for building energy optimization and climate control. Appl Energy 2018;226:1252–1272.

[7] Ren, R., Li, S.. Enhanced gaussian process regression for active learning model-based predictive control. In: CCC. 2021, p. 2731–2736.

[8] Hewing, L., Kabzan, J., Zeilinger, M.N.. Cautious model predictive control using gaussian process regression. IEEE Trans Control Syst Technol 2020;28:2736–2743.

[9] Simpson, T.W., Poplinski, J.D., Koch, P., Allen, J.K.. Metamodels for computer-based engineering design: survey and recommendations. EWC 2001;17:129–150.

[10] Liang, L., Liu, M., Martin, C., Sun, W.. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. J R Soc Interface 2018;15:20170844.

[11] Pfrommer, J., Zimmerling, C., Liu, J., Kärger, L., Henning, F., Beyerer, J.. Optimisation of manufacturing process parameters using deep neural networks as surrogate models. CIRP 2018;72:426–431.

[12] Gooijer, B.M.d., Havinga, J., Geijselaers, H., Boogaard, A.v.d.. Evaluation of pod based surrogate models of fields resulting from nonlinear fem simulations. Adv Mod Sim Eng Sci 2021;8:25.

[13] Karpatne, A., Atluri, G., Faghmous, J.H., Steinbach, M., Banerjee, A., Ganguly, A., et al. Theory-guided data science: A new paradigm for scientific discovery from data. IEEE Trans Knowl Data Eng 2017;29:2318–2331.

[14] Willard, J., Jia, X., Xu, S., Steinbach, M., Kumar, V.. Integrating scientific knowledge with machine learning for engineering and environmental systems. ACM Comput Surv 2022;55:66.

[15] Kärger, L., Bernath, A., Fritz, F., Galkin, S., Magagnato, D., Oeckerath, A., et al. Development and validation of a cae chain for unidirectional fibre reinforced composite components. Comp Struct 2015;132:350–358.

[16] Kärger, L., Galkin, S., Zimmerling, C., Dörr, D., Linden, J., Oeckerath, A., et al. Forming optimisation embedded in a cae chain to assess and enhance the structural performance of composite components. Compos Struct 2018;192:143–152.

[17] Zimmerling, C.. Machine learning algorithms for efficient process optimisation of variable geometries at the example of fabric forming. PhD-thesis at KIT; 2023.

[18] Boisse, P., Colmars, J., Hamila, N., Naouar, N., Steer, Q.. Bending and wrinkling of composite fiber preforms and prepregs. a review and new developments in the draping simulations. Comp P B 2018;141:234–249.

[19] Endruweit, A., Ermanni, P.. The in-plane permeability of sheared textiles. experimental observations and a predictive conversion model. Comp P A 2004;35(4):439–451.

[20] Zimmerling, C., Trippe, D., Fengler, B., Kärger, L.. An approach for rapid prediction of textile draping results for variable composite component geometries using deep neural networks. ESAFORM 2019;2113:020007.

[21] Zimmerling, C., Poppe, C., Stein, O., Kärger, L.. Optimisation of manufacturing process parameters for variable component geometries using reinforcement learning. Mater Des 2022;214:110423.

[22] Viisainen, J., Yu, F., Codolini, A., Chen, S., Harper, L., Sutcliffe, M.. Rapidly predicting the effect of tool geometry on the wrinkling of biaxial ncfs during composites manufacturing using a deep learning surrogate model. Comp Part B 2023;253:110536.

[23] Öchsner, A.. Composite Mechanics. Springer Cham; 2023.

[24] Kingma, D.P., Ba, J.. Adam: A method for stochastic optimization. In: ICLR. 2015, p. 1412.6980.

[25] Ronneberger, O., Fischer, P., Brox, T.. U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. Springer; 2015, p. 234–241.

[26] Lou, A., Guan, S., Loew, M.. Cfpnet-m: A light-weight encoder-decoder based network for multimodal biomedical image real-time segmentation. Comput Biol Med 2023;154:106579.

[27] Sharma, A., Kumar, D.. Classification with 2-d convolutional neural networks for breast cancer diagnosis. Sci Rep 2022;12:21857.

[28] Sharma, A., Vans, E., Shigemizu, D., Boroevich, K.A., Tsunoda, T.. Deepinsight: a methodology to transform a non-image data to an image for convolution neural network architecture. Sci Rep 2019;9:11399.

[29] Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 2004;13:600–612.

[30] Dash, T., Chitlangia, S., Ahuja, A., Srinivasan, A.. A review of some techniques for inclusion of domain-knowledge into deep neural networks. Sci Rep 2022;12:1040.

[31] Kukacka, J., Golkov, V., Cremers, D.. Regularization for deep learning: A taxonomy. CoRR 2017;cs.LG:1710.10686.

[32] Karpatne, A., Watkins, W., Read, J.S., Kumar, V.. Physics-guided neural networks (PGNN): an application in lake temperature modeling. CoRR 2017;cs.LG:1710.11431.

[33] Hoernle, N., Karampatsis, R., Belle, V., Gal, K.. Multiplexnet: towards fully satisfied logical constraints in neural networks. AAAI 2022;36:5700–5709.