Babel, Christoph; Guru, Mahish; Weiland, Jakob; Bambach, Markus

Article — Published Version

Area of interest algorithm for surface deflection areas

Journal of Intelligent Manufacturing

Provided in Cooperation with:
Springer Nature

Suggested Citation: Babel, Christoph; Guru, Mahish; Weiland, Jakob; Bambach, Markus (2024) : Area of interest algorithm for surface deflection areas, Journal of Intelligent Manufacturing, ISSN 1572-8145, Springer US, New York, NY, Vol. 36, Iss. 6, pp. 3869-3885, https://doi.org/10.1007/s10845-024-02437-9

This Version is available at:
https://hdl.handle.net/10419/323704

Journal of Intelligent Manufacturing (2025) 36:3869–3885
https://doi.org/10.1007/s10845-024-02437-9

Area of interest algorithm for surface deflection areas

Christoph Babel[1,2] · **Mahish Guru**[1,4] · **Jakob Weiland**[1] · **Markus Bambach**[3]

Received: 3 January 2024 / Accepted: 24 May 2024 / Published online: 13 June 2024

Abstract
In the automotive industry, the process of deep drawing is used for producing most of the outer surface panels. There, surface defects can occur while stamping the part. This paper proposes an area of interest (AOI) algorithm to filter possible surface deflection areas of finite element method (FEM) simulation results. The FEM is well established in the area of sheet metal forming and has shown accurate results in showing surface defects like waviness and sink marks. These two defect types are also the targeted systematic defects. In these deep drawing simulations, every manufacturing step of the sheet metal is calculated and the resulting stresses and strains are analyzed. The paper presents a newly developed post processing method for detecting surface in-corrections on basis of FEM simulation results. The focus of the method is to be independent of an experts knowledge. It should be able to be used by a wide range of non-expert applicants, unlike other post-processing methods know in today's literature. A comparison between several machine learning (ML) approaches is made. It is shown, that the developed method outperforms current state of the art approaches in terms of the recall rate. In addition, a contour tree dataset of a FEM simulation in combination with an ML approach can be successfully used to learn a multidimensional relationship between the nodes.

Keywords Machine learning · Graph neural network · Finite element method · Deep drawing · Contour tree

Introduction and state of the art

The manufacturing of most outer surface panels in the automotive industry is done by deep drawing a sheet metal with a die and a punch (Fig. 2). While the tool for deep drawing is being developed, it is very common that surface defects in the stamped parts occur. This paper will focus on two type of surface defects, waviness and sink marks (Fig. 1).

Birkert et al. (2013) state, that there are two main reasons for a systematic occurrence of such defects. The first reason is based on the designed surface geometry. If the surface curvature is small in combination with springback effects, defects become more likely. The second reason is a high stress gradient in combination with a small curvature. This can be caused by a poorly engineered tool, where too less focus is on the stress distribution in the resulting part. Both reasons can be prevented by changing the design or the manufacturing process. Because of that, simulating the production process of the part in a FEM simulation beforehand can reduce or prevent those defects.

In the deep drawing simulation, the deformation of a sheet is calculated by solving equations systems with a stiffness-matrix, a damping- and a mass-matrix (Dependant on the solving algorithm, the last two matrices are missing in the equation system). This method is well established and presented in multiple works, like (Banabic, 2010) and (Wagner, 2017). The FEM result contains deformations, strains and stresses for each discrete element. According to Weinschenk

✉ Christoph Babel
christoph.babel@bmw.de

Mahish Guru
mahishguru@gmail.com

Jakob Weiland
jakob.weiland@bmw.de

Markus Bambach
mbambach@ethz.ch

1 BMW Group, Munich, Germany

2 Chair of Mechanical Design and Manufacturing, BTU Cottbus-Senftenberg, Cottbus, Brandenburg, Germany

3 Chair of Advanced Manufacturing, ETH Zurich, Zurich, Switzerland

4 Chair of Solid Mechanics, TU Dresden, Dresden, Germany

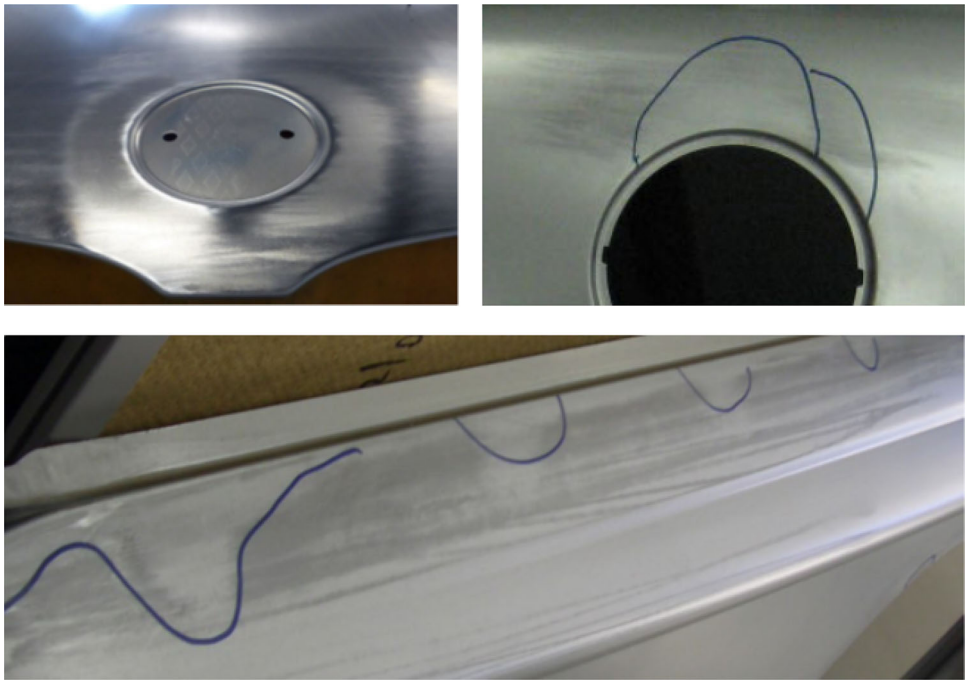


Fig. 1 Examples of the surface defect types sink mark (top) and waviness (bottom) (Babel et al., 2023)

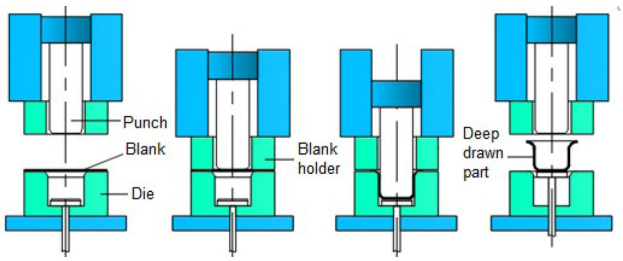


Fig. 2 Schematic deep drawing process Gürün and Karaağaç (2015)

(2020), these values are needed to predict the focused surface defects waviness and sink marks. Hartung (2000) has proven, that if the element discretization is enough, there is potential at deep drawing simulation results to show sink marks and waviness. Le Port et al. (2011) also investigated geometric systematic occuring surface defects on different parts and made the conclusion, that simulation results can have a high accuracy at showing surface defect dimensions. So in conclusion, FEM simulation results have the potential to show these type of defects.

Existing surface evaluation methods

There are many existing methods to analyze the surface of a sheet metal forming simulation result, as shown in Weinschenk (2020). The most common methods will be present in the following sections.

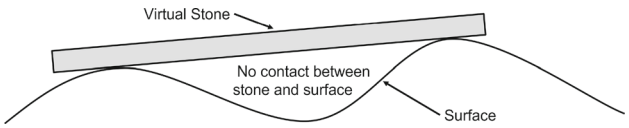


Fig. 3 Virtual stoning method with depth calculation

Virtual stoning

The virtual stoning is a very commonly used method and available in a lot of different software solutions, e.g. AutoForm (2023); LS-Dyna (2023); ESI (2023) to only name a few. The method works as shown in Fig. 3.

It is developed to imitate the stoning method in the real production environment. There, a stone of a specific length (mostly 150 mm) is used to scratch over the sheet metal in a predefined direction. Because the defective surface will have curvature changes, the stone will not scrub all the surface and geometrical defects will be visible as seen in Fig. 1. With this method it is also possible to see the geometrical dimensions of the surface imperfections. The results of this method are highly dependent on the direction of scratching and on the length of the stone. This is the main reason why an automation of this method is not sufficient.

Curvature cases

To make inhomogeneous curvature changes visible, some software products provide the user with curvature categories marked in different colors (e.g.: Fig. 4). There are different calculations of the curvature available in software products.

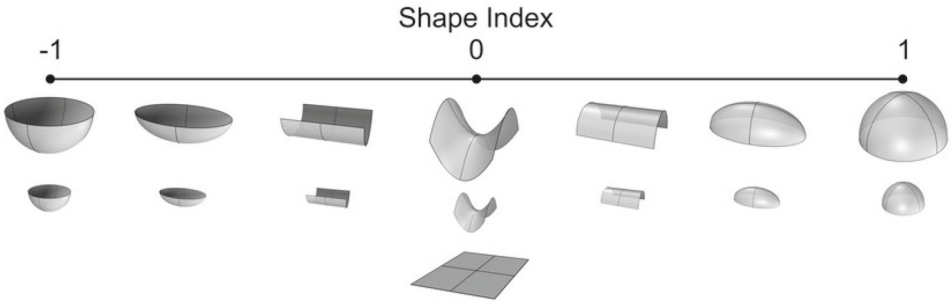


Fig. 4 Different curvature cases defined by SI and curvedness (Tsagkrasoulis et al., 2017)

The most frequently implemented curvature cases are based on the maximum and mean gaussian curvature and the principal curvatures (Zhao et al., 2013). Since these values have their downsides in making every sink mark or waviness visible, there are more complex values like e.g. the shape index (SI) (Fig. 4) (Zhao et al., 2013).

The SI is calculated from the principal curvatures k_1 and k_2, using the formula:

$$SI = \frac{1}{\pi} \cdot \arctan\left(\frac{2k_1k_2}{k_1^2 + k_2^2}\right). \quad (1)$$

The SI is a measure that characterizes the local shape of a surface at each point (see Fig. 4. It ranges between -1 and 1, where -1 indicates concave regions, 0 indicates flat regions, and 1 indicates convex regions. To map the ratio $\frac{2k_1k_2}{k_1^2+k_2^2}$ to the range $[-1, 1]$ within the domain of the arctan function, the arctan function is employed.

Negative minor strain

A negative minor strain in the simulation can result in a sink mark in the calculated part. The compressive strain forces the sheet to shrink and at areas with low geometrical stiffness this can cause a deflection after the springback simulation (Fig. 5) (Birkert et al., 2013).

To predict a possible sink mark, the user has to manually set boundaries for the maximum and minimum value of the fringe scale to get a colorized plot for the areas of interest. This method needs substantial expertise and can only be used as an indicator for surface defects.

Contour trees as preprocessing method

To get additional information for the different neural networks architectures, preprocessing steps of the provided simulation data is needed. To structure the nodal results of an FEM simulation result, one useful method is to create a contour tree out of the simulation results (Fig. 6).

Contour trees, are special form of reeb graphs, are a topological tool used for analyzing data sets that have scalar values defined on a mesh. They capture the topology of isocontours. Contour trees are useful for visualizing and analyzing complex scalar fields.

Contour trees consist of two types of nodes: extrema and joins/saddles. Extrema nodes are critical points of the scalar field where the minimum and maximum values are present (Carr et al., 2003).

The use of contour trees for preprocessing FE data is a new approach. The limitations of this method are dependant on the used data. If there is noise in the dataset, the contour tree will not generate useful representations of the data.

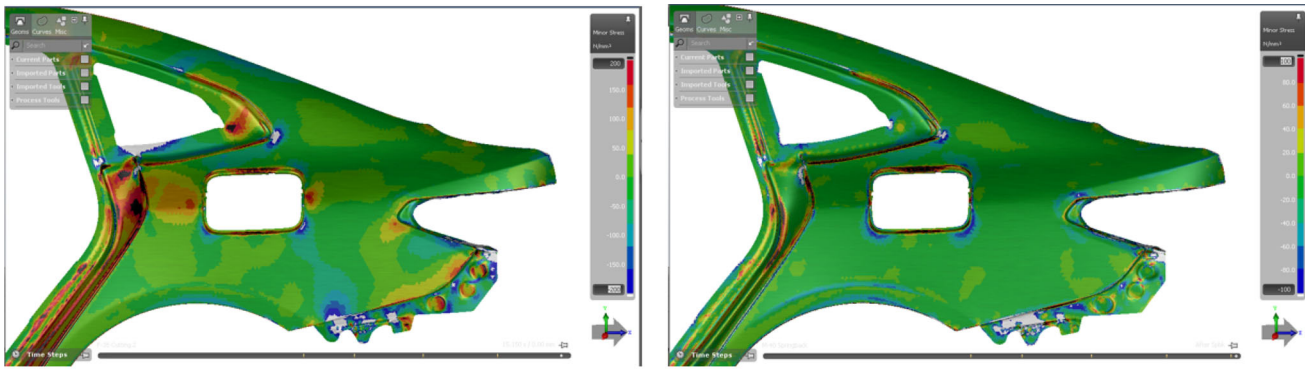
Conclusion of existing Surface Evaluation Methods

As described, every presented method has the ability to show possible areas of interests, but also its limitations. The virtual stone is a powerful tool, if the direction of scrubbing is known. The minor strain analysis is a valid tool for finding a coarse AOI, but not without additional information about the geometry, like e.g. the local topology. The curvature analysis can show complex changes in the geometry while springback, but the curvature analysis as standalone has to be analyzed by an expert because of too many diffusing information. The contour tree can help make local maxima visible, but also generates a very complex result, similar to the curvature analysis tools.

So in conclusion, every method as standalone is not sufficient for a precise AOI detection. Because of that, this paper focuses on choosing a sufficient ML algorithm to find a complex non linear connection between existing parameters to achieve a precise AOI detection.

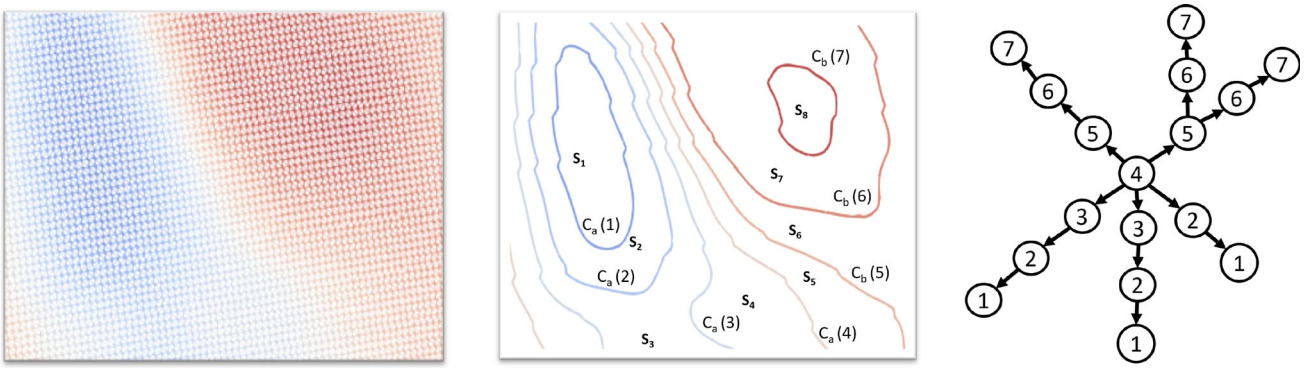
ML algorithms for AOI detection

The challenge of developing an efficient, inclusive, and precise algorithm for AOI detection can be framed as an inference problem, specifically a classification problem within a large dataset of FE variables. According to Zhang et al. (2018), classification within the context of an Function-Context-Behavior-Principle-Structure System (FCBPSS) architecture involves mapping external functional cues (as variables) to internal states (represented as 0 or 1



(a) Minor Stress before the springback simulation (b) Minor Stress after the springback simulation

Fig. 5 Comparison of minor stress before and after a springback simulation



(a) FE variable topology (b) FE variable contours (c) Contour tree

Fig. 6 Exemplary FE variable topology

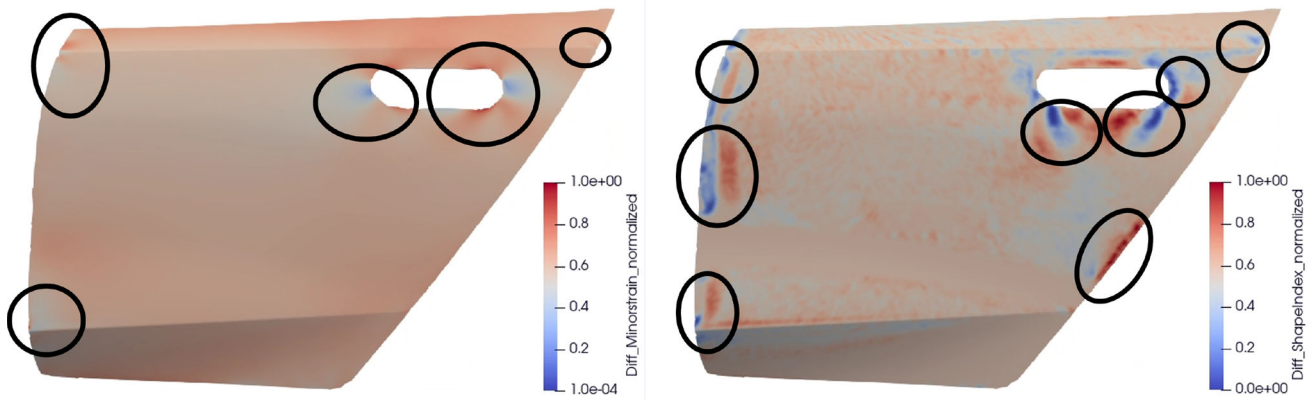
in binary classification), where the synthesis of the classification system manifests as the algorithm. In the realm of AOI detection, this classification issue follows the synthesis flow for structure (the algorithm, in this case), whereby "functional" variables are utilized to establish their typical "behavior" or pattern surrounding the AOI to create a principle for categorizing the area as either non-defective (0) or defective (1). Furthermore, it is crucial to recognize that this synthesis, guided by the principle of pattern recognition, must encompass multiple FE variables across an extensive dataset of simulation results. Deep Learning/ML models have exhibited significant efficacy in extracting meaningful features from images or videos for such tasks, as presented in Fantin Irudaya Raj & Balaji (2022). Similar methodologies have been previously employed for pattern recognition of FE variables in scenarios related to predicting force values or stress values, as mentioned in Nath et al. (2024) and Rezasefat & Hogan (2023).

The proposed method takes the nodal values (coordinates, normalized SI value, normalized difference in minor strain, etc.) of an FEM simulation into account. The relationship and influence of neighboring node values is thereby not known. Since the data is multidimensional, it is also not possible to find heuristic rules to describe the connection. For this type of problem formulation, ML algorithms are very useful (Mahesh, 2020). The selected ML approaches are already proven to work on nodal complex datasets. It will be investigated, if the algorithms of k-nearest neighbor (k-NN), random forest (RF) or graph neural networks (GNN) fit the specific use-case of this work the best.

K-NNs are widely used for classification and regression tasks. The k-NN algorithm operates by finding the k-nearest data points in the training set to a given test data point based on a distance metric, such as Euclidean distance (Cover & Hart, 1967).

RFs are an ensemble learning method based on binary decision trees. They work by creating a collection of decision trees (Bryant, 1986), each trained on a random subset of the training data and using a random subset of features (Ho, 1995).

GNNs are designed to operate on graph-structured data, where nodes represent entities, and edges capture the relationships between them (Micheli, 2009). GNNs leverage the connectivity and topology of the graph to learn mean-



(a) Normalized difference of minor strain $\widetilde{\varepsilon}_{\min}$ (b) Normalized difference of SI ($\widetilde{SI}$)

Fig. 7 Normalized difference in FE variables before and after springback

ingful representations of nodes and perform tasks such as node classification, link prediction, and graph classification. GNNs propagate information through the graph by iteratively aggregating and updating node features based on their local neighborhoods. This allows GNNs to capture complex dependencies and interactions in graph-structured data (Sanchez-Lengeling, 2023).

He et al. (2021) shows, that contour tree data in combination with a GNN can outperform existing methods in the area of hydrological datasets. In Firoze et al. (2023), the combination of contour tree data and a GNN approach is used to successfully segment tree crowns in images with an accuracy of up to 92%. In addition, Wang et al. (2021) provides a combination of a reeb graph and a GNN for classifying point cloud data with an accuracy of up to 80%.

Introduction of a new method to predict AOIs

First, the necessary input parameters and the data preprocessing will be described. Subsequently, the focus will be on the different neural network definitions.

FE variables selection

The following parameters will be analyzed if they are beneficial to use for the AOI detection:

- second principal (minor) strain $\varepsilon_{\min}$
- gradient of minor strain $|\nabla\varepsilon_{\min}|$
- hessian matrix of minor strain $|\text{Hess}(\varepsilon_{\min})|$
- principal curvatures k_1, k_2
- mean curvature H
- gauss curvature K
- SI SI

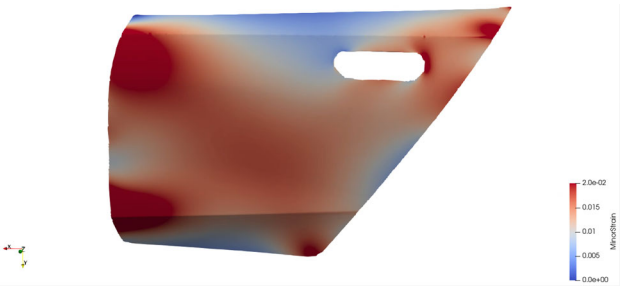


Fig. 8 Minor strain after springback

As already described in section “Negative minor strain”, a negative minor strain can indicate a surface defect. The gradient and hessian matrix will give additional information on how steep the strain is in- or decreasing (Fig. 7a).

The curvature cases are also already explained in detail in section “Curvature cases”. The principal curvatures and the gaussian curvature alone cannot indicate a sink mark or waviness. To indicate a major change in the surface topology, the parameter SI gets calculated with the principal curvature values (Fig. 7b).

All mentioned parameters will be normalized differences between the simulation result before and after springback. The difference between before and after springback is taken, because solely the springback results show insufficient results at both the minor strain values and the curvature. This can be seen e.g. between the minor strain after springback on Fig. 8 in comparison with the difference of the minor strain in Fig. 7a. The possible AOIs on this example are not detectable solely with the springback result. There is also a missing knowledge of the wanted geometry, especially when it comes to the SI. The ML algorithm does not know if a curvature change is a design feature or a manufacturing problem. With the difference in the SI, the data gets enriched with this information.

The normalization is necessary for two factors. For training a neural network, according to Sola & Sevilla (1997), a normalization improves the training speed and result. The second reason is to generalize the resulting strain values. Since there are different materials in use for outer surface panels, the usage of a specific material should not affect the outcome of the ML approach and its predictions.

Data Preparation

To provide a comprehensive overview of the data processing pipeline, a detailed summary is presented in Fig. 9.

This Figure illustrates the step-by-step process starting from the simulation results (D3plot) and the resulting shell surface (STL). It highlights the various stages of preprocessing. This includes the extraction of essential variables such as $\widetilde{\varepsilon}_{\min}$ and $\widetilde{SI}$, their topology segmentation, and the generation of the contour tree for the new variable Φ.

$$\Phi(x, y) = \frac{\overline{SI}(x, y) - \overline{SI}_{\min}}{\overline{SI}_{\max} - \overline{SI}_{\min}} + \frac{\overline{\varepsilon}_{\min}(x, y) - \overline{\varepsilon}_{\min,\min}}{\overline{\varepsilon}_{\min,\max} - \overline{\varepsilon}_{\min,\min}}, \quad \text{where} \quad (x, y) \in \Omega. \tag{2}$$

The newly introduced variable, Φ, is now utilized for topology segmentation using contour trees.

The process involves integrating data from $\widetilde{\varepsilon}_{\min}$ and $\widetilde{SI}$ into the characteristics of individual nodes, creating a Nodal dataset. This dataset is then divided into S_n parts, resulting in a Segmented dataset. Furthermore, the topological arrangement of these segments is represented using a contour tree, Γ, which is utilized to construct a graph-based dataset.

The used simulation results are all made with an element discretization of 2 mm for the sheet metal. According to Weinschenk (2020) and her experiments on sink marks, an element size between 2 mm and 4 mm shows the best comparability to experimental results. The tool speed is set to 750 mm/sec to avoid oscillation.

Labeling

The training of the networks needs labeled data. The labeling is especially in the edge areas of the defects a challenging task, since there is no fixed border of the surface errors. The goal is to mark every node and segment in an area where a defect occurs. The chosen method for labeling is a combination of visualizing the defect with a rendering software, using the segmentation images for getting the spatial limits of the deflection and taken the auditors reports of the manufactured parts into account (Fig. 10).

The combination of these different methods allows an annotation of the defective areas without confusing the networks.

To ensure a label balance between non-defective and defective nodes in the dataset, the defective nodes get oversampled so that there is a 60/40 percent balance between the two classes.

Algorithm configurations

The investigated algorithms will be RFs, k-NNs and GNNs as already described in section 1.2. The hyperparameter for the number of decision trees in the RF is set to 100, as it represents the upper limit considering all permutations and combinations of the 5 features in our attribute set X. The 5 features are as follows:

- Normalized difference of SI
- Normalized difference of minor strain
- Phi, as defined in subsection 2.2
- Normalized difference of magnitude of gradient of minor strain
- Normalized difference of magnitude of hessian matrix of minor strain

The criterion chosen to measure the quality of the split and information gain is the entropy criterion. This criterion aims to minimize entropy, which represents the average amount of information required to classify samples at a given node. The entropy criterion is suitable for learning multiple ranges of features that vary with respect to each other, making it more effective than the Gini criterion. Additionally, the minimum number of samples required to split a node is set to 2, while the minimum number of samples required to be at a leaf node is set to 1.

Similar to the RF classifier, the hyperparameters for the k-NN classifier are selected through multiple experiments to optimize precision, recall, and accuracy in binary classification for both the nodal and segmented datasets. The chosen distance metric for calculating proximity between samples in our dataset is the Manhattan distance. This metric provides the city block or taxicab distance by summing the absolute differences of the coordinates. Unlike the linear nature of Euclidean distance, Manhattan distance is more suitable for our dataset due to the presence of features with different scales and non-linear relationships among them. Therefore, we have opted for the Manhattan distance metric over Euclidean distance. The power parameter, p is set to 1 because of using manhattan distance metricz. The number of jobs paralleled to run the neighbors search is set to 4 processors.

For the GNN model, a learning rate of 1e-5 and weight decay of 5e-6 are chosen. These values control the optimization process by ensuring small and stable step sizes for weight adjustments. To enhance the model's generalization and reduce reliance on specific features, a dropout

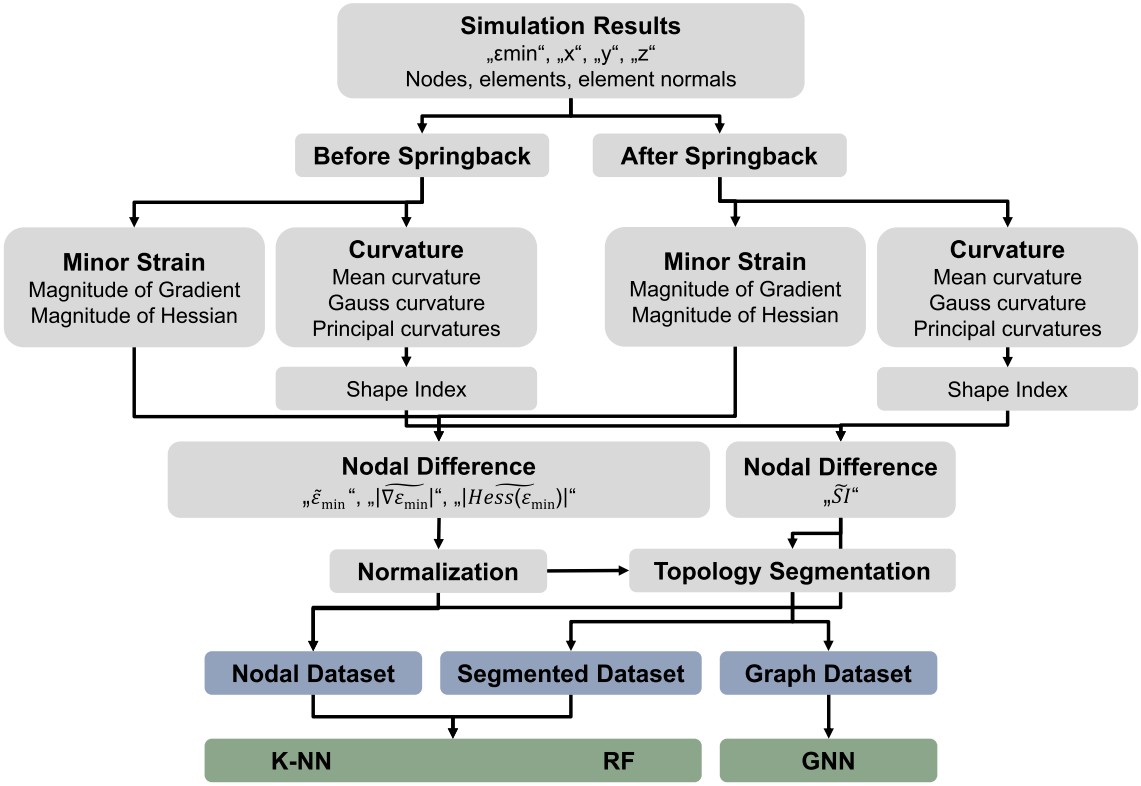
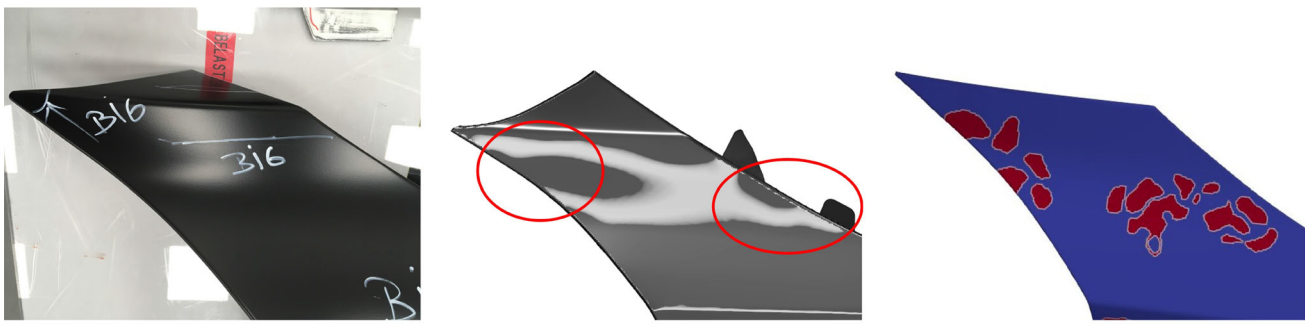


Fig. 9 Data processing pipeline for the AOI detection



(a) Auditor's report (b) Light line rendering analysis (c) Annotations

Fig. 10 Labeling of the AOI dataset

ratio of 0.50 is applied. This technique helps prevent overfitting and promotes the overall versatility of the network. In order to prevent bias towards any particular class during weight training, a batch size of 40,000 nodes is selected. This choice ensures proportional representation of each class and enables the model to learn effectively from both majority and minority classes. To address the challenge of class imbalance present in the graph dataset, class weighing is implemented in the batch loader of the data loader.

Experiments and results

In this section, the experiments on the different network architectures will be described including the presentation of the main results. The dataset contains simulation results of five fenders and five doors of different car projects with over 140 defective areas and approximately 7000 nodes.

The focus is on examining different feature combinations and their respective performances in the context of RF and k-NN classifiers. Investigations are made for the following feature combinations for both RF and k-NN:

Table 1 Input data overview for the different experiments

Classifier	Input data				Data type			
	$\overline{\varepsilon}_{min}$, $\overline{SI}$	$\overline{\varepsilon}_{min}$, $\overline{\Phi}$, $\overline{SI}$	$\widetilde{\varepsilon}_{min}$, $\widetilde{\Phi}$, $\widetilde{SI}$	$\widetilde{\Phi}$, $\widetilde{SI}$, $\widetilde{\varepsilon}_{min}$, $\widetilde{\lvert\nabla\varepsilon_{min}\rvert}$, $\widetilde{\lvert Hess(\varepsilon_{min})\rvert}$	NOD	NOV	SEG	SOV
K-NN	X	X	X	X	X	X	X	X
RF	X	X	X	X	X	X	X	X
Chebconv				X			X	X
GAT				X			X	X

1. $\overline{\varepsilon}_{min}$, $\overline{SI}$: In the context of the multistage forming simulation, analysis is made for the nodal differences in minor strain and SI before and after the springback stage.
2. $\overline{\Phi}$, $\overline{\varepsilon}_{min}$, $\overline{SI}$: Incorporation of the additional computed nodal difference of Φ variable in the feature set for the analysis alongside the minor strain and SI.
3. $\widetilde{\Phi}$, $\widetilde{\varepsilon}_{min}$, $\widetilde{SI}$: To account for different material properties and deformation characteristics of various automotive panels, the previous feature set is normalized, creating a new feature set that enables a more comprehensive analysis.
4. $\widetilde{\Phi}$, $\widetilde{SI}$, $\widetilde{\varepsilon}_{min}$, $\widetilde{|\nabla\varepsilon_{min}|}$, $\widetilde{|Hess(\varepsilon_{min})|}$: In the final enhancement of the feature set, the normalized magnitude of the gradient and Hessian matrix of minor strains is incorporated.

The enhancement on the feature set with the normalized magnitude of the gradient and the normalized magnitude of the hessian matrix of the minor strain provides additional semi-local information.

The nodal minor strain data discussed above lacks local information on the variations of minor strain values in neighboring nodes. However, the surrounding area of a given node exhibits distinct characteristics regarding minor strain variations, especially in defective regions. To capture this crucial information, calculating the gradient of $\varepsilon_{\min}^{(N)}$ becomes necessary. This gradient calculation helps capture the changes in minor strain in the three directions. Subsequently, by considering the magnitude of this gradient,

$$|\nabla\varepsilon_{\min}^{(N)}| = \sqrt{\left(\frac{\partial\varepsilon_{\min}^{(N)}}{\partial x}\right)^2 + \left(\frac{\partial\varepsilon_{\min}^{(N)}}{\partial y}\right)^2}, \tag{3}$$

quantitative assessment of $\varepsilon_{\min}^{(N)}$ at each node's neighborhood.

Furthermore, the magnitude of the Hessian matrix of $\varepsilon_{\min}^{(N)}$,

$$|\mathrm{Hess}(\varepsilon_{\min}^{(N)})| = \sqrt{\left(\frac{\partial^2\varepsilon_{\min}^{(N)}}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2\varepsilon_{\min}^{(N)}}{\partial x\partial y}\right)^2 + \left(\frac{\partial^2\varepsilon_{\min}^{(N)}}{\partial y^2}\right)^2}. \tag{4}$$

Table 2 Overview of chosen hyperparameters for the DOEs

Classifier	Hyperparameter			
	K	Depth	Hidden layers	Hidden dimension
K-NN	1–100			
RF		1–40		
Chebconv			1–20	6,10,12,16
GAT			1–20	6,10,12,16

indicates the extent of variation in the magnitude of the gradient of $\varepsilon_{\min}^{(N)}$. Incorporation of this in the data gives additional neighborhood information for each node and enhances the information captured.

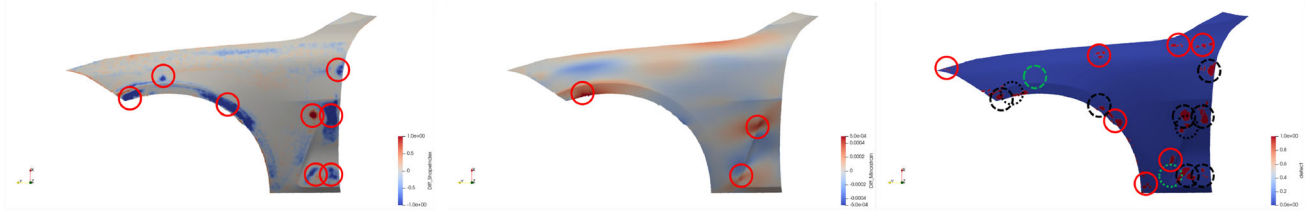
The experiments on the three different ML approaches focus on different hyperparameters. An overview is given in Table 1, where NOD stands for the nodal dataset, NOV for the nodal oversampled dataset, SEG for the segmented dataset and SOV for the oversampled segmented dataset.

Table 2 shows an overview of the different tested hyperparameter for the Design of Experiments (DOE).

The training results get validated by a separate test set, that contains new additional data, which are not present in the training data set.

Analysis with state of the art process

To ensure that the performance of the developed method can be compared with the currently established methods, the 10 parts get manually inspected with the analysis of minor strain values and the curvature cases as described in section "Curvature cases" and "Negative minor strain". Figure 11 shows the analysis of one exemplary part. The areas with a high difference in shape and minor strain are marked and are compared with the ground truth areas previously defined in section "Labeling". If a ground truth AOI is not covered by a manually marked area, it is counted as False Negative Area. If a manually marked area is in a place where no ground truth defect occurs, it is counted as False Positive Area. The marking of the area is not nodewise, since the exact boundaries are not clearly detectable with the two used methods. The virtual stoning method is excluded, since this analysis needs



(a) Difference in SI (b) Difference in Minor Strain (c) Labeled Ground Truth (Red: False Negative, Green: False Positive, Black: True Positive)

Fig. 11 Example of a manual marking on an FE result

a pre knowledge of possible AOIs and is then applied with a certain scrubbing direction.

The analysis with the existing methods show an overall recall of 0,46 and a precision of 0,78. Many areas are difficult to detect, because the fringe bar range has to be adapted multiple times to get a clear image where a potential AOI is located.

RF algorithm

RF experiments are conducted using a DOE approach to investigate the impact of the maximum depth hyperparameter on the classifier's performance. The maximum depth values range from 0 to 50, while keeping other parameters constant.

Figure 12 shows the experiments on different depths and variables selection. The inclusion of the additionally computed parameter phi leads to improvements in all three parameters recall, precision and accuracy. Also, the normalized variables perform better which underlines the need for normalizing the strain values. Because of those results shown in 12 the following experiments are done with the final enhancement of the feature set.

The next experiments are focusing on the selected data format. It is investigated, if an oversampling and an addition of segmented data helps the RF classifier to perform better regarding the precision, recall and accuracy. Figure 13 shows the results.

It is clearly visible, that the nodal dataset perform better with focusing on the recall at depths between 10 and 50. On the other hand, the segmented dataset has a better accuracy and precision at lower network depths. Because of the significant difference between the precision values of nodal and segmented data, the preference is on using only nodal data at the RF classifier. Also, there is a major difference at the recall with the oversampled nodal dataset. Therefore, the oversampled dataset get preferred.

Another experiment is made to analyze the splitting of the dataset into training, validation and testing set. Since there are 10 different parts used for the dataset, it is possible to split the dataset also in 7 parts for training, 2 parts for validating and 1 part for testing. The other dataset is split randomly to 70%, 20%, and 10%. The results in Fig. 14 are showing that the random split performs better, suggesting not enough heterogeneous data in the validation and testing set when splitting by parts.

K-NN algorithm

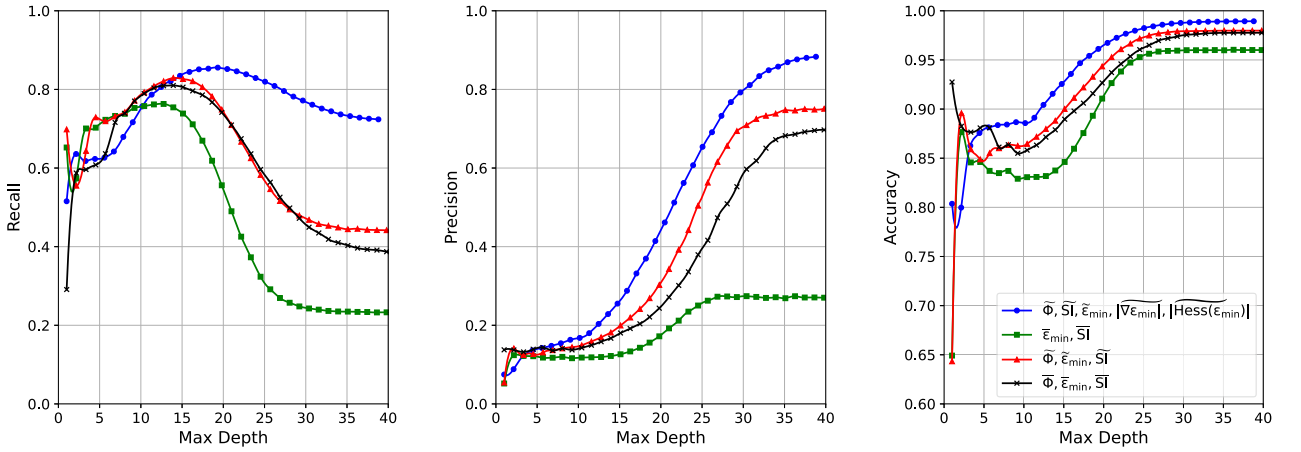
The main hyperparameter for the k-NN classifier is k, which represents the number of neighbors considered for classification. The choice of the optimal value for k is crucial, as it impacts the bias-variance trade-off. A small value of k may lead to overfitting, while a large value may result in underfitting. Therefore, a DOE approach is utilized to explore the effect of multiple values of k ranging from 0 to 300 on their effectiveness in AOI detection.

The experiments with the k-NN are the same as with the RF classifier. The results of the k-NN are also showing, that the enhanced dataset performs best and that the random split improves the results on basis of the recall, precision and accuracy the most (Figs. 19 and 20). One major difference in comparison to the RF is the oversampling strategy result (Fig. 15).

This behavior can be attributed to the increase in the number of neighbors when using a higher value of k. As the number of neighbors increases, the KNN classifier considers a larger neighborhood around each data point, including more samples near the AOI. This leads to a higher number of true positives, resulting in improved recall and precision rates for the oversampled datasets. However, it is crucial to also consider the corresponding accuracy values. The trade-off between the increase in recall and the decrease in accuracy becomes apparent in the oversampled dataset.

GNNs

The number of hidden layers is a crucial hyperparameter in GNNs as it determines the depth of neighborhood representation learning around each node in the graph dataset. Additionally, the number of hidden dimensions is essential as



(a) Recall vs Maximum depth (b) Precision vs Maximum depth (c) Accuracy vs Maximum depth

Fig. 12 RF's performance of various feature combinations (using Recall, Precision and Accuracy rate vs Maximum depth)

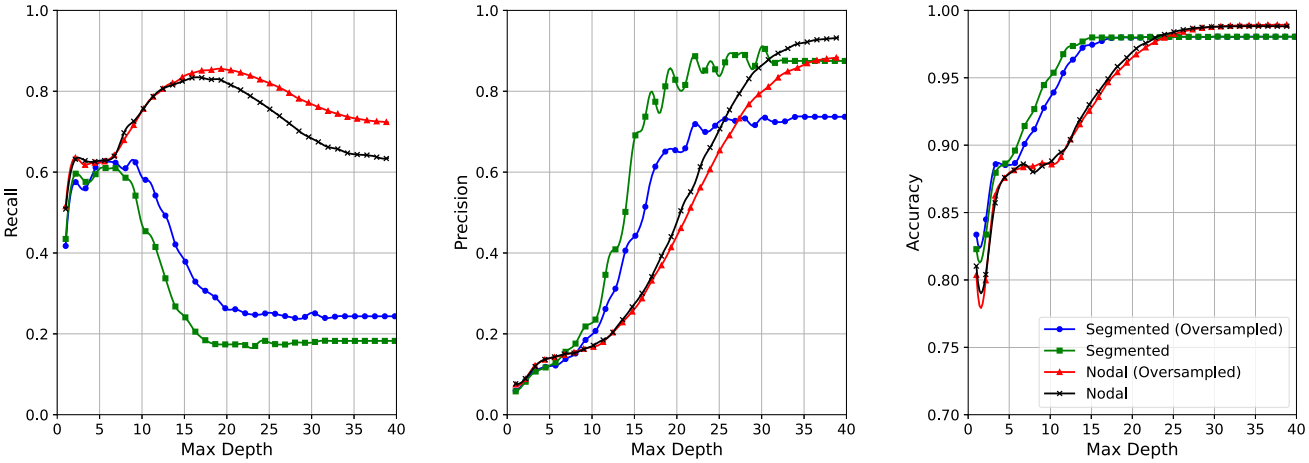


Fig. 13 RF's performance on nodal and segmented data, with and without an oversampling strategy

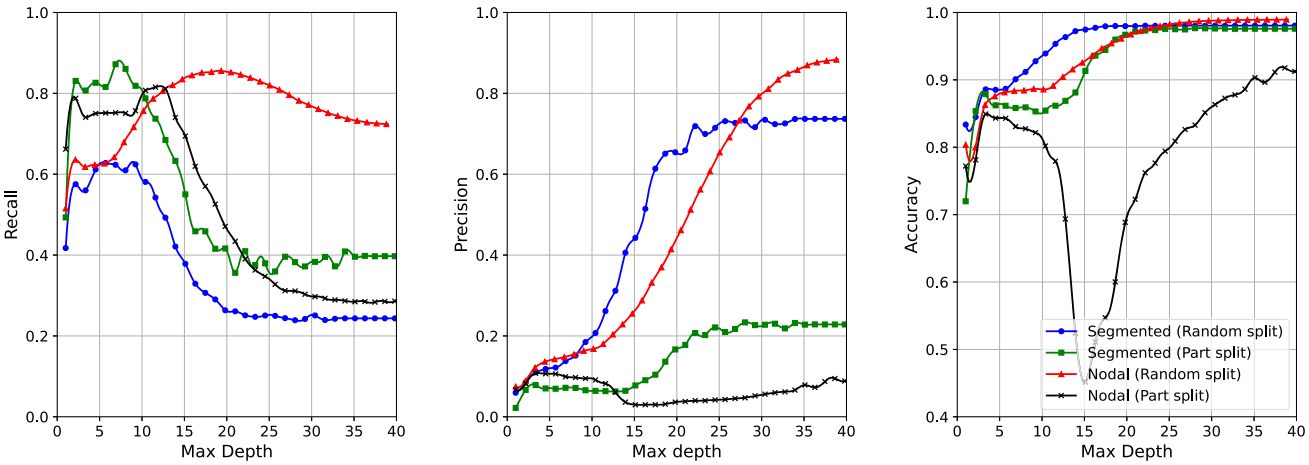
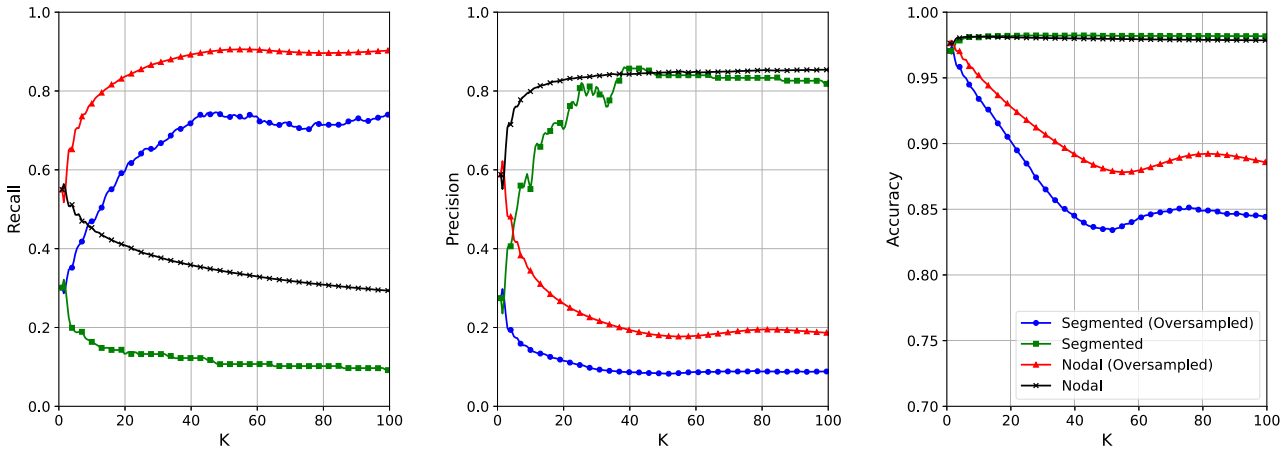


Fig. 14 RF's performance comparison of random data split and part data split



(a) Recall vs K (nearest neighbors) (b) Precision vs K (nearest neighbors) (c) Accuracy vs K (nearest neighbors)

Fig. 15 K-NN's performance of nodal and segmented dataset, with and without oversampling

it specifies the dimensionality of the hidden space required to efficiently capture non-linear relationships between features. Multiple values of hidden layers and hidden dimensions are explored to evaluate their impact on the binary classification performance of GNNs. DOE techniques are employed to investigate the number of hidden layers ranging from 1 to 20 and the investigated numbers of hidden dimensions are 6, 10, 12 and 16.

Another significant hyperparameter specific to GNNs is the selection of the convolutional operator. Different convolutional operators have distinct advantages and disadvantages in terms of message passing and aggregation from neighboring nodes.

- Chebconv, a spectral graph convolutional framework that can effectively capture both local and global information from the graph data. The perfect balance of local and global data would be very advantageous for our contour tree-graph dataset.
- Graph Attention Networks (GAT) enable the model to assign varying importance to neighborhood nodes, focusing only on nodes with the most relevant information for classification. This is particularly advantageous for the Graph dataset.

Therefore, a range of Chebyshev filters in Chebconv and a range of attention heads for GAT are investigated to assess their performance in AOI detection.

The experiments on the Chebconv are showing the most promising results for 16 hidden dimensions and six hidden layers (Fig. 16).

While the accuracy increases with more hidden layers, the recall value drops below an acceptable range.

The experiments on the GAT are shown in Fig. 17. The comparison of different hidden dimensions in the GAT architecture reveals interesting patterns in recall and accuracy rates for AOI classification.

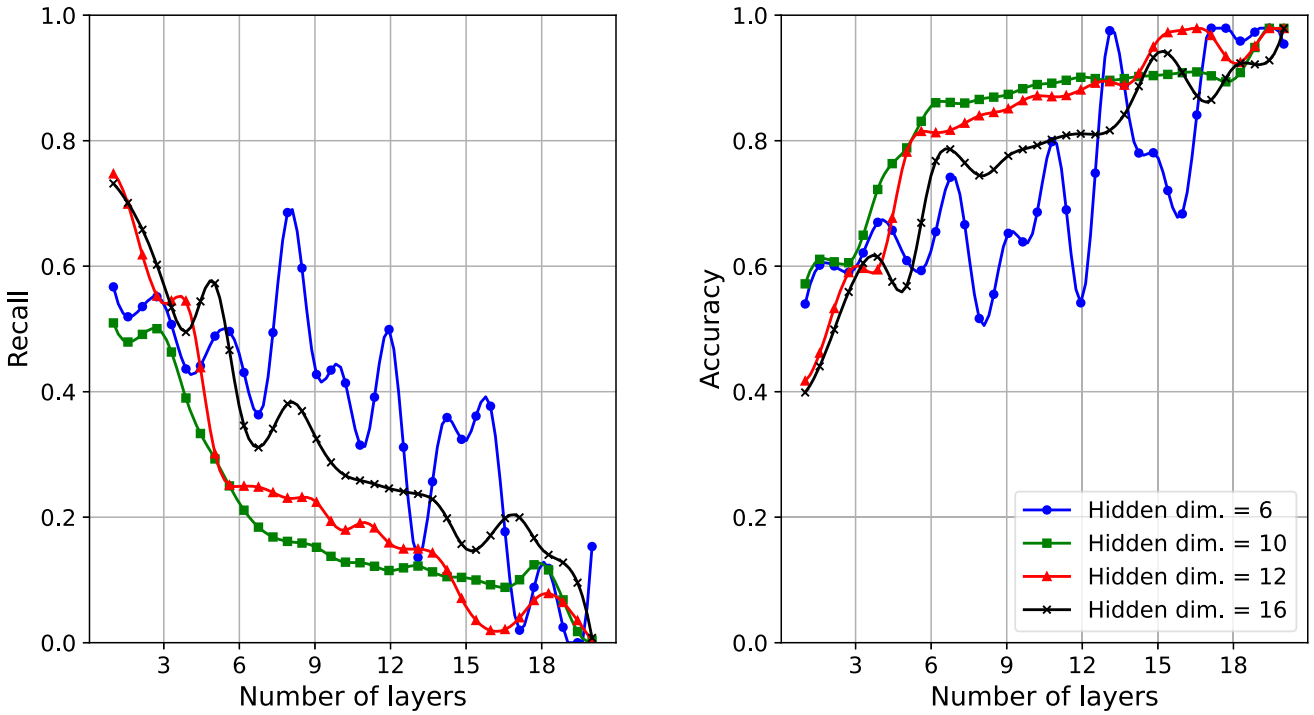
For the hidden dimension of 6, significant oscillations are observed which indicates that the model may be underfitting due to insufficient resolution in capturing the dataset's inherent dimensionality. The lack of data to adequately learn the underlying patterns required for accurate AOI classification may be contributing to this behavior.

Furthermore, the oscillations in the recall and accuracy rate curves tend to reduce as the number of hidden dimensions increases, particularly for 10 and 12 hidden dimensions, and for 10–20 number of hidden layers. This behavior can be attributed to the limited amount of data available for training and learning within these architectures, which leads to fluctuations in performance metrics.

The most stable performance in both recall and accuracy rates is observed for the higher hidden dimension of 16. This suggests that a higher hidden dimension allows the GAT architecture to better represent and learn the underlying features in the data, resulting in more consistent and reliable AOI classification.

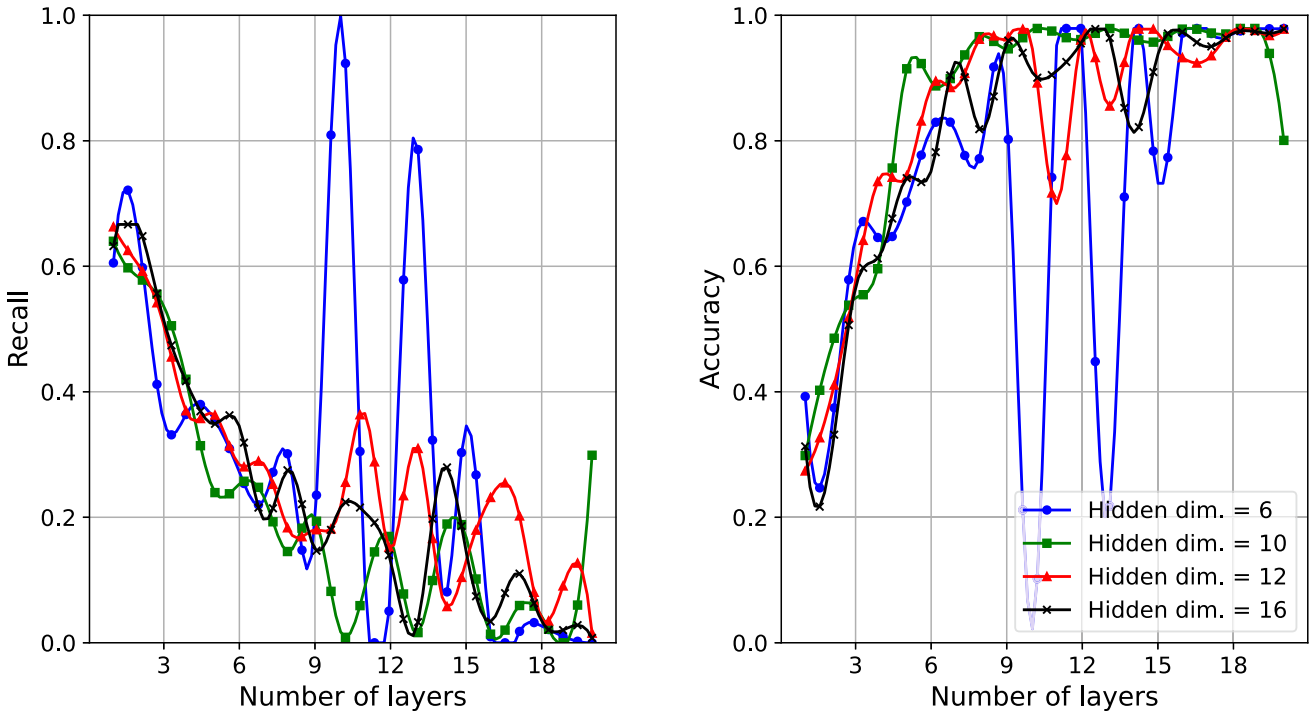
However, it is worth noting that, despite achieving higher accuracies, the low recall rates imply that the GAT with 16 hidden dimensions and 8 attention heads detects very few true positive nodes. This highlights the need for more data to effectively identify all patterns and optimize the model's hyperparameters.

In conclusion, the GAT has shown promise for improving AOI classification performance. However, the limited data availability has posed challenges in effectively capturing underlying patterns. Further research and experimentation are needed to obtain a larger dataset to fully explore the



(a) Recall vs Num. of hidden layers (b) Accuracy vs Num. of hidden layers

Fig. 16 Chebconv's performance with different hidden dimensions



(a) Recall vs Num. of hidden layers (b) Accuracy vs Num. of hidden layers

Fig. 17 GAT's performance with different hidden dimensions

potential of the GAT. Additionally, hyperparameter tuning is crucial to optimize the model's performance. Parameters such as dropout ratio, batch sizing, and post-message passing layers can significantly impact the model's accuracy and recall rates. Careful and extensive tuning of these parameters is necessary to achieve the best possible results in AOI classification.

Comparison

Table 3 presents a comprehensive comparison of the performance metrics achieved by various classifiers, including RF, k-NN, Chebconv and GAT, for AOI classification. The best-performing metrics were determined based on the investigations conducted in the previous subsections.

To ensure an objective comparison, all classifiers in their best performing modes (according to best balance in advantage between accuracy and recall) were trained using the part split technique. There, 7 parts were allocated for training, 2 parts for validation, and 1 part for testing. The inference of AOIs in the test set was performed using all the other classifiers. For this comparison, a rear door outer panel served as the basis for making inferences on the overall 2D shell mesh domain, denoted as Ω_{test}. The table includes Accuracy and Recall rates along with the corresponding confusion matrix, illustrating the performance on the outer rear door panel.

Furthermore, Fig. 18 visually represents the corresponding AOI classification performances, where the red areas are classified as AOI with label 1 (positive label), and the blue areas are classified with label 0 (negative labels). This comprehensive analysis allows for a clear and objective evaluation of the different classifiers' performance in AOI classification, aiding in understanding their strengths and weaknesses in detecting and localizing areas of interest.

The best-performing classifier, with an accuracy of 0.8679 and a recall rate of 0.8630, is the RF classifier. The corresponding inference results can be visualized in Fig. 18b, and a detailed representation of the performance is provided in the form of a confusion matrix in Table 3.

In the confusion matrix, it is evident that there are numerous true negatives graph nodes in the inference, indicating that the classifier accurately identifies a lot of non-defective areas as negatives. Additionally, there are many true positive labels correctly classified as positive. However, it is observed that some of the negative-labeled graph nodes are incorrectly considered as positive, though this proportion is relatively small compared to the true negatives.

This type of performance, while not perfect, is generally acceptable for AOI detection. Although there are some misclassifications, the RF classifier achieves a high overall accuracy and recall rate, successfully capturing a significant portion of the defective areas while maintaining a reasonable level of false positives.

The inference of k-NN classifier, as depicted in Fig. 18c, demonstrates decent AOI classification in this particular part during inference. However, upon examination of the recall rate and the true positive section from the corresponding confusion matrix in the table, it becomes evident that there are many positive graph nodes in the ground truth that are not correctly classified as positive in the inference. Additionally, several negative graph nodes from the ground truth are incorrectly labeled as positive.

This behavior is also reflected in the figure, where some missed true positives (false negatives) and some false positives are observed. These misclassifications indicate that the k-NN classifier might not be as effective in capturing all the true positive areas, leading to both missed defect detections and false alarms in non-defective regions. While the k-NN classifier exhibits decent AOI classification performance, the relatively lower recall rate and misclassifications suggest that it may not be as reliable as the RF classifier. The RF classifier showed in comparison higher accuracy and recall rates.

Figures 18d and 18e, along with the corresponding metrics in Table 3, illustrate the inference performance of the test part using the GNN classifiers. In these figures, the GNN classifiers, namely Chebconv and GAT, perform AOI classification on the test part. Additionally, Table 3 provides detailed performance metrics for each GNN classifier, including accuracy and recall rates, along with the confusion matrix. These metrics offer valuable insights into the GNN classifiers' abilities to detect and localize areas of interest in the test part.

The Chebconv convolutional architecture exhibits a low recall rate of 0.2739, indicating that only a small proportion of positive ground truth graph nodes are correctly labeled as positive. However, the overall accuracy is high at 0.98. This high accuracy implies that there are relatively few graph nodes classified as positive, whether true or false.

Similarly, the best performing mode of the GAT convolution architecture also lacks a significant number of positive graph node inferences, true or false, despite its improved recall rate compared to Chebconv. The GAT convolution figure shows that some important AOIs are correctly labeled, but overall, the number of positive inferences remains limited.

The primary challenge for both GNN architectures is the class imbalance problem. The dataset lacks sufficient representation of positive nodes in the ground truth. This makes it challenging for the classifiers to effectively learn patterns in the message passing around positive graph nodes. The class weighting employed in the loss function of all the GNN networks, in the last layer of post message passing, is not sufficient to maintain dataset balance.

Table 3 Comparison of all the classifiers' metrics for AOI detection

Classifiers	Accuracy	Recall	Precision	Confusion Matrix		*Predicted label* 0	*Predicted label* 1
RF	0.8679	0.8630	0.075	*True label*	*0*	5036	766
					1	10	63
K-NN	0.8347	0.5616	0.041	*True label*	*0*	4863	939
					1	32	41
GNN - Chebconv	0.9826	0.2739	0.28	*True label*	*0*	5754	49
					1	53	20
GNN - GAT	0.9766	0.3287	0.217	*True label*	*0*	5715	88
					1	49	24

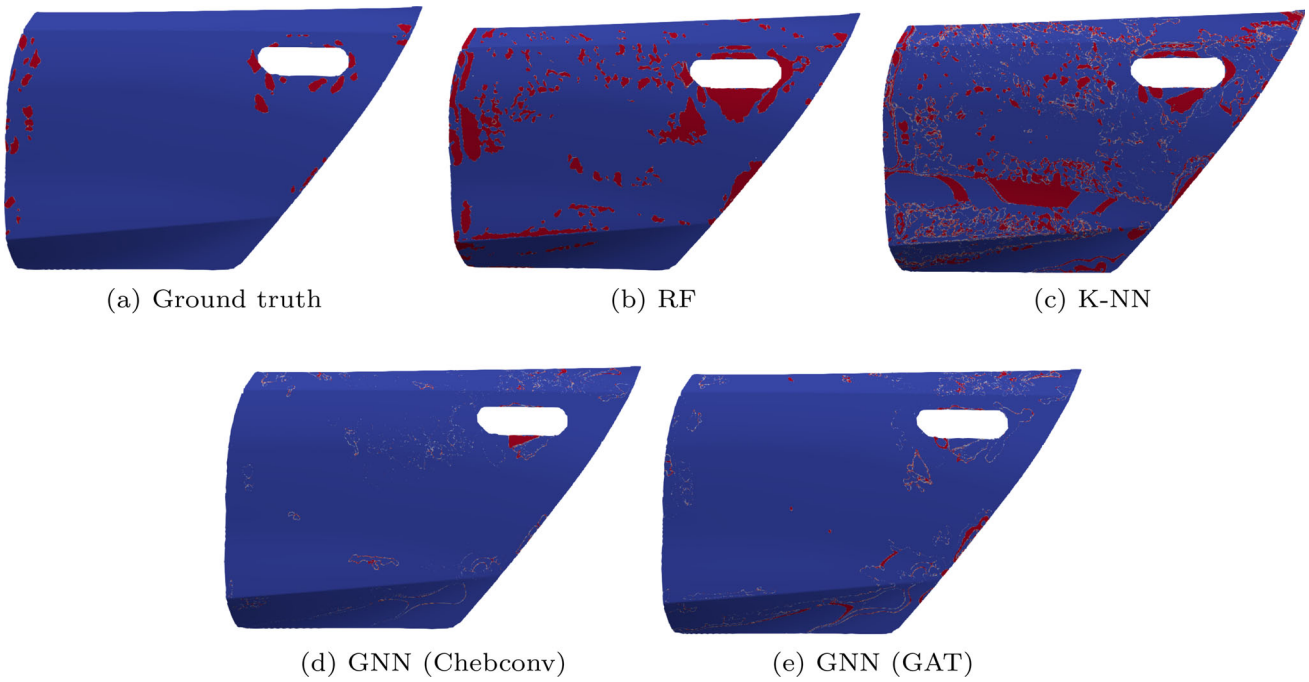


Fig. 18 Comparison of all the classifiers for AOI detection against the ground truth AOIs for an outer automotive door panel

Conclusion

This research proves, that it is possible to generate an usable surface defect detection method without the need of expert knowledge. The proposed method introduces a novel data processing and data enhancement strategy to transfer from a manual AOI detection to an automated approach. The strategy involves enhancing the data by incorporating not only nodal information but also local information. Through various operations (eg. gradient calculation and contour tree generation) on the minor strain and curvature data, datasets can be generated that contain localized and neighborhood information regarding the topologies of these variables. This strategy quantitatively enriches the dataset for the development of an automated algorithm for AOI classification, making the process significantly more manageable and effective.

This research shows the viability of utilizing minor strain and curvature variables obtained from FEM forming simulations before and after springback to identify an area as a defective AOI with the help of an ML approach. It shows the potential of different ML approaches, including RF, k-NN, and various GNN architectures. These methods are utilized to automatically discover patterns in the enhanced dataset, which aids in classifying local neighborhoods as defect-prone or not.

In conclusion, this research demonstrates that RF classifiers applied to the enhanced contour tree segmented dataset outperform all other classifiers. Even though the results do not show perfect effectiveness because of a relatively low precision, it outperforms the state of the art method in terms of sensitivity. It detects not every AOI of the ground truth, but detects all the more important AOIs in addition with extra areas.

Furthermore, the research shows that GNNs, specifically the GAT network with post message passing layers showcased a very good accuracy, but had very low recall and precision rates. The results confirm the successful use of FEM contour tree dataset in combination with an ML approach. It proves, that it's possible to learn a multidimensional relationship between nodal values of the FEM result.

Future work

As previously discussed, the current dataset lacks in sufficient representation for a GNN to effectively detect patterns related to characteristic AOIs. To address this limitation, one promising approach is to increase the number of datapoints of the labeled dataset. By doing so, the class imbalances can be mitigated, providing neural networks with more diverse and representative samples from each category. This augmentation will enable the model to gain a better understanding of distinguishing defective and non-defective regions.

As presented in Joloudari et al. (2022) on Convolutional Neural Networks (CNNs), the Synthetic Minority Oversampling Technique (SMOTE) can be beneficial for achieving a better training result. This method can also be investigated to the present dataset, if the performance on precision and recall rates can be further improved.

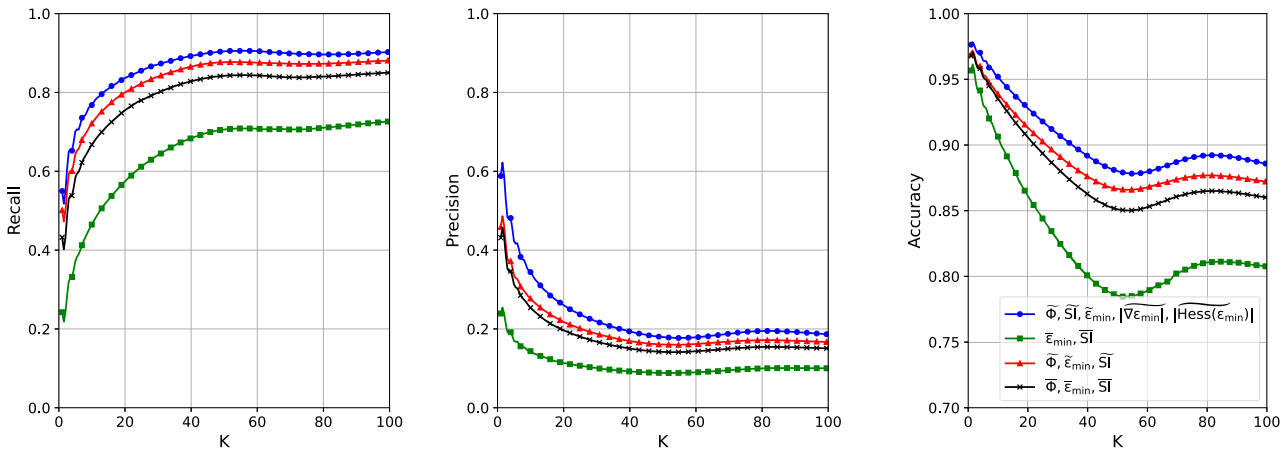
Moreover, fine-tuning the hyperparameters of the GNNs can lead to further improvements in performance when applied on imbalanced datasets. It can help to optimize the GNN's ability to capture the nuances of AOIs and enhance its overall detection accuracy.

By employing such tuning strategies, significant enhancement could be seen in the performance of GNNs in AOI detection, resulting in more accurate and reliable defect identification in various applications.

Regarding the dataset and investigated variables, the addition of other variables can be proved, if it leads to a better precision of defining an area as defective. Possible variables with additional information can be e.g. the curvedness factor and the major strain. In addition, it can also be investigated, if the defined variable ϕ for generating the contour tree can be further improved.

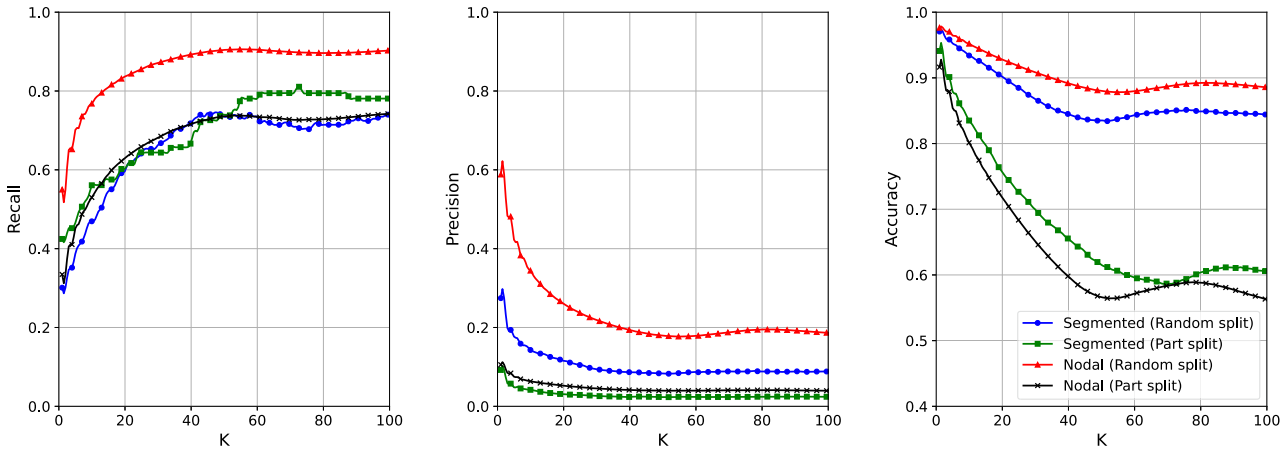
Appendix A: Additional figures

See Appendix Figs. 19 and 20



(a) Recall vs K (nearest neighbors) (b) Precision vs K (nearest neighbors) (c) Accuracy vs K (nearest neighbors)

Fig. 19 kNN's performance of various feature combinations (using Recall, Precision and Accuracy rate vs K (nearest neighbors))



(a) Recall vs K (nearest neighbors) (b) Precision vs K (nearest neighbors) (c) Accuracy vs K (nearest neighbors)

Fig. 20 kNN's performance comparison of random data split and part data split

Funding Open Access funding enabled and organized by Projekt DEAL.

References

AutoForm. (2023). AutoForm Manual. AutoForm Engineering GmbH.

Babel, C., Weiland, J., & Bambach, M. (2023). Introduction of a method for systematic surface defect classification on virtual car body parts. In: Proceedings of the 2023 10th International Conference on Industrial Engineering and Applications, pp 295–301.

Banabic, D. (2010). Sheet metal forming processes: constitutive modelling and numerical simulation. Springer Science & Business Media.

Birkert, A., Haage, S., & Straub, M. (2013). *Verfahrenstechnische Grundlagen der Karosserieteilherstellung* (pp. 195–316). Berlin Heidelberg, Berlin, Heidelberg: Springer.

Bryant, R. E. (1986). Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers, 100*(8), 677–691.

Carr, H., Snoeyink, J., & Axen, U. (2003). Computing contour trees in all dimensions. *Computational Geometry, 24*(2), 75–94.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory, 13*(1), 21–27.

ESI. (2023). Pam-Stamp 2020.0.1 - Reference Manual. ESI Group AG.

Fantin Irudaya Raj, E., & Balaji, M. (2022). *Application of Deep Learning and Machine Learning in Pattern Recognition* (pp. 63–89). Singapore: Springer Singapore.

Firoze, A., Wingren, C., & Yeh, R.A., et al. (2023). Tree instance segmentation with temporal contour graph. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 2193–2202.

Gürün, H., & Karaağaç, I. (2015). The experimental investigation of effects of multiple parameters on the formability of the dc01 sheet metal material. Strojniški vestnik - Journal of Mechanical Engineering 61.

Hartung, C. (2000). Beurteilung des optischen erscheinungsbildes von ziehteilen mit hilfe numerischer verfahren. PhD thesis, Technical University Munich, Munich.

He, W., Sainju, A.M., & Jiang, Z., et al. (2021). Deep neural network for 3d surface segmentation based on contour tree hierarchy. In: Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), SIAM, pp 253–261.

Ho, T.K. (1995). Random decision forests. In: Proceedings of 3rd international conference on document analysis and recognition, IEEE, pp 278–282.

Joloudari, J.H., Marefat, A., & Nematollahi, M.A., et al. (2022). Effective class-imbalance learning based on smote and convolutional neural networks.

Le Port, A., Thuillier, S., Borot, C., et al. (2011). Analysis, Simulation and Prediction of Cosmetic Defects on Automotive External Panel. *AIP Conference Proceedings, 1383*(1), 228–236.

LS-Dyna. (2023). LS-Dyna Manual R13.0 Vol I. Livermore Software Technology Corporation

Mahesh, B. (2020). Machine learning algorithms-a review. International Journal of Science and Research (IJSR)[Internet] 9(1):381–386

Micheli, A. (2009). Neural network for graphs: A contextual constructive approach. *IEEE Transactions on Neural Networks, 20*(3), 498–511.

Nath, D., Neog, D.R., & Gautam, S.S., et al. (2024). Application of machine learning and deep learning in finite element analysis: A comprehensive review. Archives of Computational Methods in Engineering pp 1–40

Rezasefat, M., & Hogan, J. D. (2023). A finite element-convolutional neural network model (fe-cnn) for stress field analysis around arbi-

trary inclusions. *Machine Learning: Science and Technology, 4*(4), 045052.
Sanchez-Lengeling, B. (2023). A gentle introduction to graph neural networks. https://distill.pub/2021/gnn-intro/.
Sola, J., & Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on nuclear science, 44*(3), 1464–1468.
Tsagkrasoulis, D., Hysi, P., Spector, T., et al. (2017). Heritability maps of human face morphology through large-scale automated three-dimensional phenotyping. *Scientific reports, 7*(1), 45885.
Wagner, M. (2017). *Lineare und nichtlineare FEM*. Springer.
Wang, W., You, Y., Liu, W., et al. (2021). Point cloud classification with deep normalized reeb graph convolution. *Image and Vision Computing, 106*, 104092.
Weinschenk, A. (2020). Simulative und experimentelle untersuchungen zur detektion und praevention von einfallstellen in außenhautbauteilen. PhD thesis, Technical University Munich, Munich.
Zhang, W., Yang, G., & Lin, Y., et al. (2018). On definition of deep learning. In: 2018 World Automation Congress (WAC), pp 1–5.
Zhao, H., Anwer, N., & Bourdet, P. (2013). Curvature-based registration and segmentation for multisensor coordinate metrology. *Procedia CIRP, 10*, 112–118.