Journal of Manufacturing Processes 145 (2025) 571–580

Contents lists available at ScienceDirect

Journal of Manufacturing Processes

journal homepage: www.elsevier.com/locate/manpro

Graph Neural Networks for patterned welds detection on point clouds

Kojo Welbeck [a], Yahui Zhang [a], Ru Yang [a], Guangze Li [b], Hui-Ping Wang [b], Blair Carlson [b], Ping Guo [a,*]

[a] Department of Mechanical Engineering, Northwestern University, Evanston, IL, 60208, USA
[b] Materials & Manufacturing Systems Research Lab, General Motors, Warren, MI 48092, USA

ARTICLE INFO

Keywords:
Point cloud
Laser welding
Graph Neural Network
Machine vision inspection
Instance and semantic segmentation
Weld detection

ABSTRACT

Remote laser welding is an increasingly common process in automotive assembly, particularly in body-in-white and battery manufacturing. While 3D vision cameras enable quick-turnaround inspections of the welds, traditional machine vision algorithms to detect welds, namely template-matching raster-search implementations, often struggle with the variability in weld appearance and quality. Consequently, these implementations may fail to detect partial welds, distorted welds, or welds with any significant deviation from the templates used within the algorithms. Also, the raster scan sometimes detects false positives, welds that are, in fact, not present. To address these challenges, this study introduces a weld surface detection system that uses Graph Neural Networks (GNNs) to detect the presence and locations of welds in 3D point cloud data, implicitly incorporating the variability inherent in the training data samples. The proposed deep-learning algorithm comprises two GNN models chained together: one for segmenting welds by the direction and identity, and another for locating the center of welds. This approach also operates directly on point clouds, offering a computationally efficient alternative to the typical 2D normal map projection or 3D voxelization pre-processing operations on point clouds. In the experimental results, the proposed algorithm identified all welds present, including those with shape deviations, demonstrating a relative strength compared to a template-matching baseline currently used in production.

1. Introduction

Laser welding is widely employed in the automotive industry for joining both structural and non-structural components [1,2]. Increasingly, laser welding equipment is integrated into robot cells as part of end-of-arm tooling, where robotic path tracing enables repeatability of the weld shapes, sizes, and locations [3]. Additional fixturing on the parts being welded together provides a datum for the target weld locations, enhancing the repeatability of welds across different instances of sub-assemblies that pass through automated welding cells or stations [4]. Yet, even with consistent fixturing and preprogrammed robotic paths, there can still be significant variability in the weld formation, with quality implications for the welds themselves and, by extension, for the vehicle. The high energy of the lasers used in welding processes creates dynamic and non-equilibrium reactions [5], leading to variations in surface quality across welds. Individual welds, thus, have to be inspected to ensure quality [6]. In particular, welds are evaluated on their morphological features, and are deemed structurally sound when these features meet predefined threshold standards [7–9].

3D point clouds of welds (Figs. 1(a) and 1(b)) enable surface quality assessments of the welds' morphologies. Before inspection, a detection step is required, which is the focus of this study. The objective of detecting welds from such scans can be categorized as an object detection task on point clouds, which also has relevance in various fields beyond manufacturing [10]. Traditional template-matching implementations on 3D point clouds commonly involve organizing the point clouds into a structured format before searching using a predefined template [11]. Point clouds are often organized into grid-based structures, such as by computing 2D normal maps (Fig. 1(c)), projecting and averaging them into 2D images, or voxelizing them into 3D grids [12].

During template-matching, manually-crafted kernels or templates operate on these grid structures by convolution [13]. Incidentally, the existing traditional weld-detection program, featuring manually-crafted kernels for template-matching, failed to detect some welds, and, in some cases, detected false positives (Fig. 2). The failure modes of the current system prompted an investigation into a deep learning alternative to handle the responsibility of locating welds. To wit, in 2D or 3D Convolutional Neural Networks (CNNs) [14], the convolution kernels can be learned directly from data, which theoretically leads to more representative templates, thus improving the accuracy of object detection by template-matching.

* Corresponding author.
E-mail address: ping.guo@northwestern.edu (P. Guo).

https://doi.org/10.1016/j.jmapro.2025.03.109
Received 26 March 2025; Accepted 28 March 2025
Available online 30 April 2025

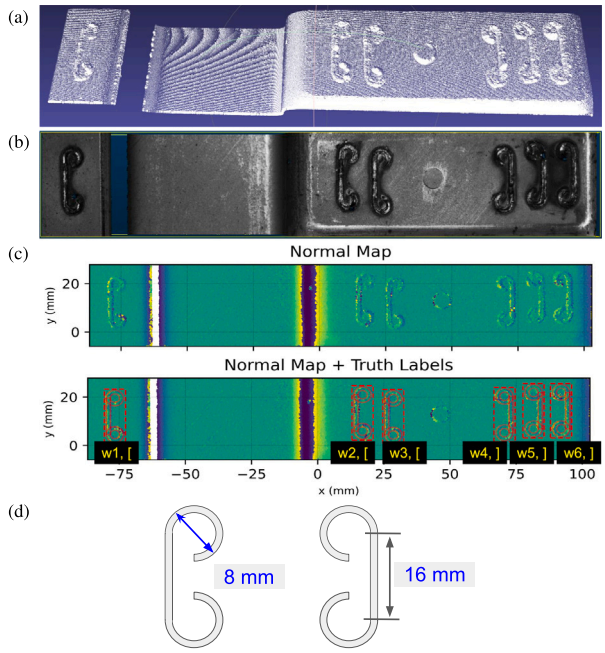


Fig. 1. (a) Off-axis view of a 3D scanned section of the floorpan, with 6 staple welds; (b) Gray-scale top-view of the intensities from the same point cloud scan; (c) 2D normal map of the point cloud scan, with ground truth weld locations overlaid and additional tags to identify the weld directions and weld numbers; (d) Illustration of the left-facing (left) and right-facing (right) staple welds.

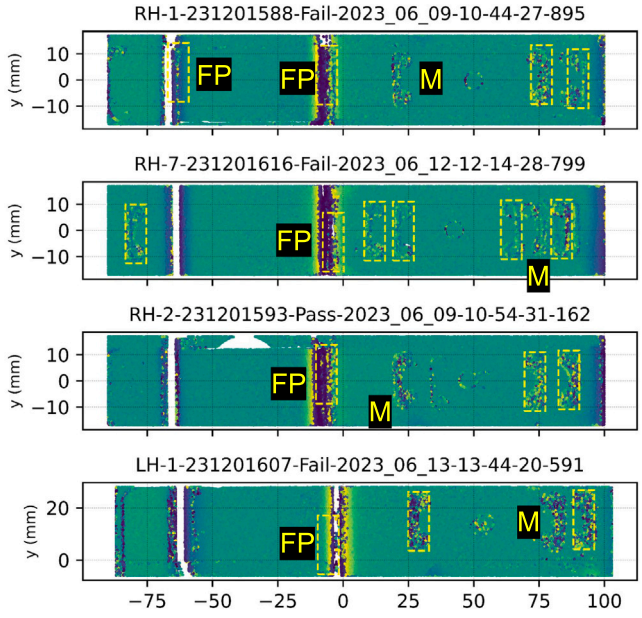


Fig. 2. Examples of misclassified welds based on the traditional template-matching. 'FP' indicates the false positive, while 'M' indicates the missed weld.

However, point clouds are inherently unstructured and sparse, attributes that do not naturally favor grid structures. The aforementioned grid-based conversions or projections require additional computations of varying complexity, which often leads to a loss of information from the original point cloud and to the introduction of artificial artifacts or inaccuracies. Conversely, graph-based methods allow for various levels of processing unstructured point clouds in their sparse 3D representations [15]. By encoding point clouds as graphs, which inherently capture high-fidelity spatial proximity relationships, we circumvent the need to project or voxelize point clouds into 2D or 3D grid structures, and avoid losing or obscuring some raw information of the original data. Therefore, Graph Neural Networks (GNNs) offer distinct advantages over CNNs and other grid-based methods when processing point clouds as graphs. The GNN-based algorithm presented in this study combines elements of PointNet [16], PointNet++ [17], and PointGNN [15] to effectively detect, segment, and locate the welds from point cloud scans.

PointNet [16] demonstrates the use of multi-layer perceptrons (MLPs) to extract point embeddings from point cloud coordinates. The architecture leverages MLPs in conjunction with permutation-invariant pooling operations, like max-pooling, to approximate continuous single value functions that operate on sets of points. PointNet++ [17] builds upon PointNet by introducing a density-informed adaptive sampling of the cloud; and uses PointNets hierarchically to balance expanding the receptive fields that help to capture more spatial context, while maintaining computational efficiency. Finally, PointGNN [15] operates on point clouds by first casting them into a graph structure. It generates and updates node-, edge-, and graph-embeddings along the network using the graph casting as a computation graph, then terminates with point-wise classifications and bounding-box predictions. PointGNN, in short, is an end-to-end network that is trained to detect objects of interest from a point cloud.

We draw inspiration from these methods to learn weld-detection from labeled samples. This paper details a specific implementation for processing 3D point cloud scans with a relatively large field of view to pinpoint the locations of welds of a specific geometric shape. The specific weld shape in this treatment is that of a staple: two arc segments vertically aligned and joined by a vertical line segment on either the left or right ends of the arcs (Fig. 1(d)). The staple welds are patterned across the floorpans of the GMC Hummer EV, and samples of these (Fig. 1(a)) were collected to train the GNNs. The GNN-based algorithm has been validated on staple welds in this paper and can be generalized to detect other parametric shapes as well. This paper specifically addresses a low-cost scenario where only the location information of points is available in the point cloud. The proposed method achieves robust performance without relying on point normals, gray-scale intensities, color information, or accompanying 2D images from the vision system.

In summary, **(1)** we propose and investigate a new algorithm to detect the location of staple-shaped welds from a point cloud scan that includes multiple welds. The algorithm chains together two GNNs, one for identifying welds in the scan and a second for predicting the location of each identified weld. The sequential split affords some high-level explainability of the welds and their detections, and provides two lever for tuning incremental improvements on the whole task. This algorithm is specifically scoped to apply deep learning techniques to the permutation-invariant unstructured format of point clouds, and to limited resolution point clouds that contain only location information of constituent points.

(2) We compare the results of the GNN-based network to a traditional baseline algorithm that uses pattern matching on a 2D normal map of the scan. The proposed algorithm demonstrates clear robustness against missed detections and false positives, which are common issues with the pattern-matching approach.

The detailed algorithm, network structure, and training procedures are first described in Section 2. Section 3 presents the intermediary and summary results from processing the evaluation datasets and discusses some noticeable sources of error. Section 4 suggests possible avenues for improvement, to close out.

2. Methodology

Compared to PointGNN [15], where the objects of interest within the point cloud are varied and may have complex structures, the objects in this study are parametrically defined staple shapes (Fig. 1(d)),

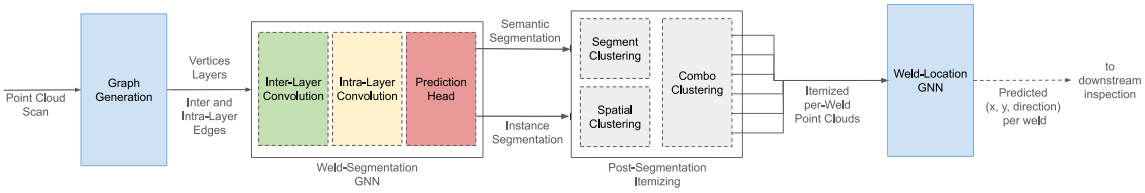


Fig. 3. An overview of the serial connection of two separate GNNs for weld detection and localization, and the flow of information from scan to inspection.

created with robotic precision. Furthermore, the detection pipeline, which identifies the presence, direction, and identity of welds within a large area scan, is implemented using two separate GNNs in series, as summarized in Fig. 3, rather than a single end-to-end detection network. The decomposition allows for analyzing intermediate states of the process, and back-tracing for the sources of anomalous detection predictions, which equates to more explainability and control than would otherwise be obvious.

The first GNN serves as a segmentation network, classifying points as belonging to specific welds with a specific facing direction (Fig. 1(c)) or as background points of the floorpan, effectively distinguishing weld-related members from the surrounding background. The second GNN is a location regression network, predicting a weld's location from all the points previously adjudged to belong to that particular weld. A weld itemizing (or weld extraction) procedure bridges the two networks. After the three-step sequence of identifying, extracting, and locating welds within the point cloud, the predicted locations are forwarded to a pre-existing inspection algorithm for evaluation. This final step determines whether the welds meet quality standards or require repair. Lastly in comparison to PointGNN, the proposed pipeline operates on higher-resolution stereo-correlation scans, compared to the wider-volume lower-resolution LIDAR scans in the PointGNN experiments. This detection task expects sub-millimeter accuracy in the weld location predictions. The following subsections detail the respective stages of the sequence.

2.1. Weld-segmentation GNN

The first-stage GNN functions as both a semantic and instance segmentation network, where segmentation involves classifying each point individually then aggregating all classifications. In the semantic segmentation task, each point is labeled based on weld direction—whether the weld is facing left, facing right, or is part of the background. In the instance segmentation task, each point is assigned a weld number, indicating whether it belongs to (from left to right) the first weld, second weld, and so forth, or is part of the background. This approach determines the number of welds present, their point members, and the welds' orientations by member consensus. The weld orientation is necessary for the downstream inspection system.

This Weld-Segmentation network stage adopts the approach used in PointGNN, starting by casting the scanned point cloud into a graph structure of high-resolution and low-resolution node layers. It incorporates an auto-registration mechanism to learn corrective adjustments to individual node coordinates, and extracts node embeddings from coordinates and edge embeddings from displacement vectors. Additionally, it follows PointNet++'s hierarchical point set abstractions to selectively transfer information from the high-resolution points to the low-resolution points used for predictions. Subsequently, the network processes these low-resolution points through an interplay of multi-layer perceptrons (MLPs) and max-pooling operations to generate the segmentation output, similar to PointGNN. However, unlike PointGNN, the final prediction head in this network classifies points along two axes—weld direction and weld number—omitting the bounding-box location and dimension predictions.

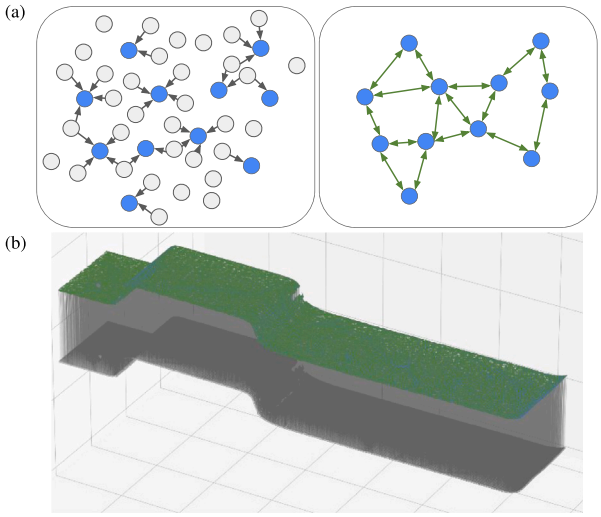


Fig. 4. Graph Generation for Weld-Segmentation: (a) illustration of the two layers of graph vertices, V_1 (light gray circles) and V_2 (blue circles), with the inter-layer edges (dark gray arrows) and intra-layer edges (green arrows); and (b) example of the two layers of graph vertices with the inter-layer edges (gray lines) and intra-layer edges (green lines).

2.1.1. Graph generation for weld-segmentation

The scanned point cloud is organized into collections of vertices (or nodes) (V) and edges (E), representing the graph, $G(V, E)$. The graph dictates the computation graph of the GNN, instructing the network on which nodes will share information by message passing and aggregation. The graph construction features two sets (or layers) of nodes: an initial high-resolution set (V_1) and the down-sampled lower-resolution set (V_2) on which predictions will be based. Down-sampling reduces the number of computations and was necessary to fit the full graph on the GPU available. Each point cloud is already pre-down-sampled from the original scan resolution of 0.08 mm to the 0.2 mm "high-res" V_1 layer. This initial resizing is to replicate the case of having to work with lower-quality 3D scanners. The final lower-quality set is a further re-sampling to a 0.6 mm resolution (V_2). A collection of inter-layer edges, E_{inter}, pass messages between the two sets (V_1) and (V_2), and a collection of intra-layer edges, E_{intra}, share information across the nodes of the low-res layer, all as part of the network architecture described in the next subsection. Fig. 4(a) illustrates and 4(b) shows an example of the graph structures, defined by:

$$\begin{aligned} G(V, E) &= G(V_1, V_2, E_{inter}, E_{intra}) \\ E_{inter} &= \{\ (i,j)\ \mid\ \|V_2[j] - V_1[i]\| \le 0.4\,\text{mm}\ \} \\ E_{intra} &= \{\ (i,j)\ \mid\ \|V_2[j] - V_1[i]\| \le 1.2\,\text{mm}\ \}. \end{aligned} \tag{1}$$

2.1.2. Weld-segmentation network architecture

The network is constructed as a sequence of three sub-networks: the inter-layer convolution sub-net, the intra-layer convolution sub-net and the prediction head. The operational details of the sub-networks are illustrated in Fig. 5. The max-pooling operation, an example of a permutation-invariant aggregation function, handles the message passing between nodes via edges, by collecting the highest values for each

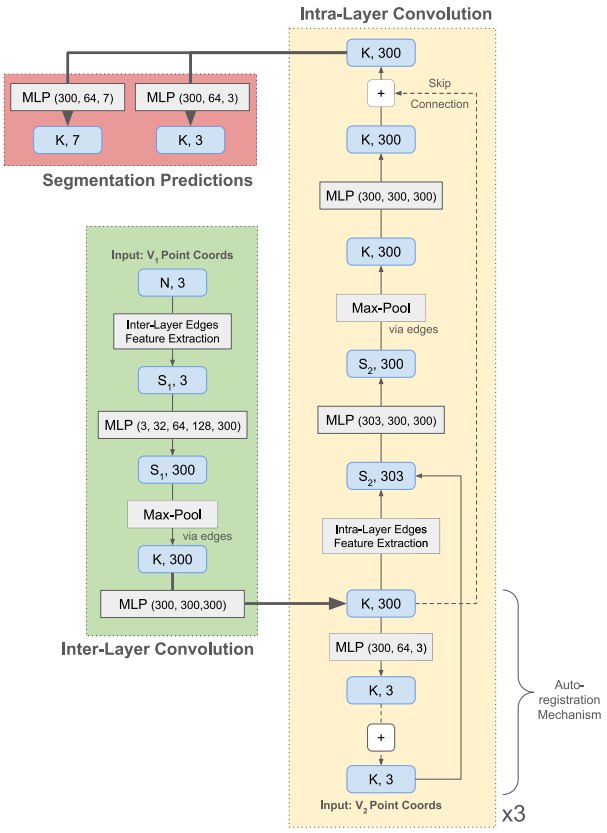


Fig. 5. Weld-Segmentation network, featuring message passing and aggregation from the high-resolution layer of vertices to the lower resolution down-sampled layer of vertices using the Inter-Layer Convolution subnet, then across the vertices of the low-resolution layer through their respective edge embeddings using the Intra-Layer Convolution subnet.

dimension across all edge embeddings with a shared destination [18]. The output of this operation is a new node embedding for each destination vertex. Pooling operations are followed by MLPs to update the pooled node embeddings. The prediction head sub-network classifies each point by weld direction and weld number, by attributing to each class the point's probabilities of belonging to a given direction and a given weld instance.

2.1.3. Dataset for weld-segmentation

The dataset of 602 point clouds was collected by scanning weld-containing sections of the floorpans of 28 GMC Hummer EVs during production. Each scan was recorded using LMI Technologies' Gocator 3210 to capture 190 mm × 34 mm sections of the floorpan by stereo-correlation, and at a resolution of 0.08 mm. A corresponding label text file was created for each scan, to list the 2D location and weld-direction for each of the 3 to 6 welds present in the scan. All labels were manually verified. The per-weld locations and directions recorded in the label files were translated into point-wise weld-direction and weld-number labels before processing samples through the model. This translation was executed by, first, superimposing a staple shape mask on the point cloud at the ground truth weld locations. The masked points were then labeled according to their respective weld numbers and weld directions. Thus, our training point clouds had now been augmented to include two additional ground truth attributes per point, alongside their 3D locations: the direction of the weld and the number or identity of the weld that the point belongs to.

2.1.4. Objective function featuring focal loss cross entropy

The classification objective was implemented with the Focal Loss [19] adaptation to the Categorical Cross-Entropy Loss function. The

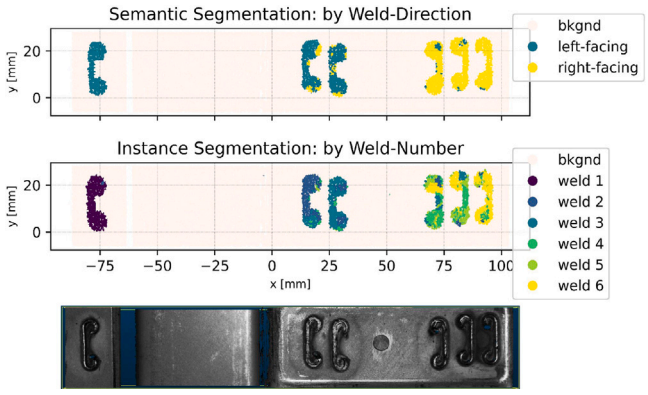


Fig. 6. Semantic and instance segmentation prediction plots from the Weld-Segmentation GNN on an example scan. Each point of the downsampling is color-coded by weld-direction (top) and by weld-number (middle). The gray-scale top-view image of the point cloud intensities is pictured (bottom) for reference.

Categorical Cross-Entropy loss calculates a measure of uncertainty or entropy for each category which is then summed across all samples in the training set. Focal Loss extends the cross-entropy loss by attaching more weight to the entropies of the hard-to-classify points. The α-adjusted variant [19] of Focal Loss, used in this training, scales these atomic losses to correct for the imbalance in the representations across the different classes. In this case, background nodes make up an exaggerated majority of the scan, so their losses will be scaled down according to their outsized representation:

$$\text{Focal Loss} = \sum_{i}^{N} (1 - p_c)^{\gamma} \; log(p_c) \; \alpha_c \mid t(i) = c$$
$$\alpha_c = \Big(\sum_{i}^{N} (1 \mid t(i) = c)\Big)^{-1}, \tag{2}$$

where $t(i)$ is the truth label for the ith node; p_c is the predicted probability of belonging to class c; and γ is the focusing parameter that controls by how much to down-weight easy samples.

The objective function (3) incorporates a Focal Loss term for both the semantic ($\text{Focal}_{semantic}$) and instance ($\text{Focal}_{instance}$) classification subtasks, placing greater emphasis on the instance segmentation task. Additionally, a regularization term on the model parameters was included to temper down weight values and overfitting:

$$\text{Loss} = \text{Focal}_{semantic} + 10 \times \text{Focal}_{instance} + \sum_{\forall w_k \in W} \|w_k\|, \tag{3}$$

where W is the set of all model parameters.

2.1.5. Weld-segmentation training

The network was trained on 481 random samples from the full set of 602 scans, in a 5-fold cross-validation experiment. The graphs were constructed with less than the full complement of available edges, as in dropout regularization [20]. Each model was trained stochastically using PyTorch's stock Adam optimizer [21] with a 0.0001 learning rate. The output predictions of the segmentation network on an example scan are shown in Fig. 6. Generally, the semantic segmentation predictions are more accurate compared to the instance segmentation. However, both results contain scatterings of misclassified points. A summary of results across the test scans is provided in the results Section 3.1.

2.2. Post-segmentation itemizing

Once the segmentation network has highlighted the presence of welds, a post-segmentation step is required to extract subsets of the

point cloud corresponding to each identified weld—referred to as itemized weld clouds. Due to imperfections in the segmentation results, extracting accurately shaped welds can be challenging, which makes post-segmentation clean-up necessary.

Three different post-process solutions have been tested: a "Spatial-Clustering" scheme, a "Segment-Clustering" scheme, and a "Combo-Clustering" scheme. Each scheme, described in the subsections that follow, incorporates the Density-Based Spatial Clustering Application with Noise (DBSCAN) algorithm [22] to itemize and sanitize individual welds.

2.2.1. Density Based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN [22] is an unsupervised technique to separate a set of data points into regions of dense clusters of points, agnositic to cluster shape. The algorithm is employed in the subsequent clustering schemes to identify welds as the densest contiguous regions of segmented points from the point cloud, and to remove clusters that are too small as noise. The DBSCAN algorithm finds core points of high density and expands clusters out from those core points. Two global parameters govern the algorithm's results: the neighborhood radius and the minimum density threshold, which were set to 0.2 mm and 20, respectively, in these experiments.

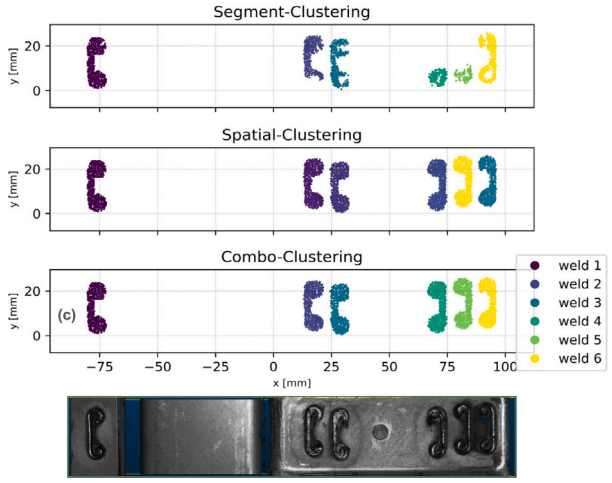
2.2.2. Spatial-clustering

Spatial-Clustering (1) removes all background-predicted points aggregated as the union of background classifications in the semantic and instance segmentation tasks; (2) clusters the remaining points in 3D space with DBSCAN, and retains the largest clusters of points as welds, above a certain size threshold; then (3) assigns weld-directions and weld-numbers to each weld by majority vote on the clusters' semantic and instance predictions.

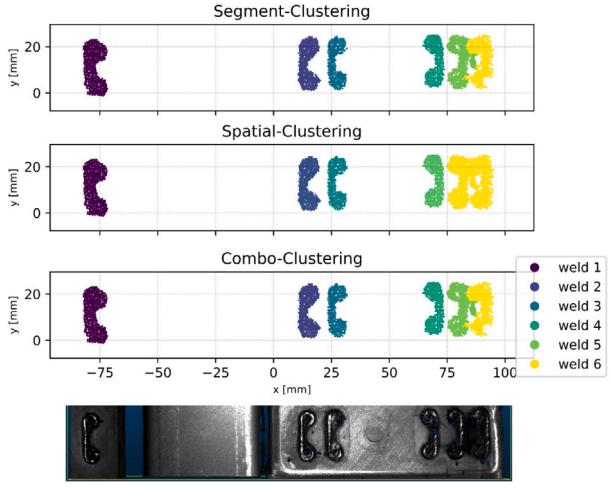
Spatial-Clustering performs its best when the welds are well spaced out. In these cases, weld clusters retain the general staple shape of welds, like with the masking used in point-wise labeling. In the frequent occurrence of multiple welds in very close proximity to one another, what should be separate welds are clustered as a larger conglomerate. Clustering multiple welds as one merged weld negatively affects the clean up, and propagates through the rest of the weld-detector algorithm as missed welds and weld detections with large errors in the predicted locations.

2.2.3. Segment-clustering

Segment-Clustering (1) similarly removes the background-predicted points, post-segmentation; (2) counts the number of predictions for each weld number (1 to 6) and filters the counts through a size threshold to identify the number of welds to detect; (3) extracts the points predicted as belonging to each weld instance, counting up to the number of welds identified, then with DBSCAN, excludes all points but the largest contiguous cluster as the weld; then finally, (4) with the weld-number already assigned from the instance segmentation predictions, weld-directions are deduced by majority vote on the clusters' semantic segmentation predictions. Segment-Clustering, for the most part, avoids the merged clusters problem of spatial-clustering, except for when a contiguous cluster from the noisy segmentation predictions bleeds into other welds. A consistent artifact of Segment-Clustering is that extracted welds do not hold to the general shape of staple welds, and by varying degrees. The large variability of cluster shapes negatively impacts the accuracy of the downstream weld-location network.



(a) The three post-segmentation clustering schemes executed on a sample scan that has been processed by the first-stage Weld-Segmentation network. Segment-Clustering does not always yield staple-shaped clusters.



(b) Combo-Clustering retains the itemization from Spatial-Clustering for welds 1–4, and defers to the Segment-Clustering detections for welds 5 and 6, avoiding a case of, and the effects of, merged welds.

Fig. 7. Demonstrating the post-segmentation clustering schemes on two sample scans, where Combo-Clustering fuses the results from Segment-Clustering and Spatial-Clustering.

2.2.4. Combo-clustering

Combo-Clustering compares the itemized welds of Spatial-Clustering and Segment-Clustering. In the cases where Spatial-Clustering erroneously merges two welds, combo-clustering favors the itemization from Segment-Clustering.

The result of the clustering clean-up is a collection of subsets of points, each subset corresponding to a weld detected in the original scan. Each itemized weld cloud of the collection is individually forwarded to the next GNN. An example of the post-segmentation clean-up and itemization results on the instance segmentation output of a sample scan is plotted in Fig. 7(a), according to each of the three clustering schemes. Combo-Clustering's ability to differentiate and correct merged welds is specifically demonstrated in Fig. 7(b).

2.3. Weld-location GNN

Weld location inference is handled by a regression network that processes each itemized weld cloud into a graph-wise prediction of the weld's 2D center location. By this point, the weld's direction has already been determined by consensus during the preceding post-segmentation

clustering cleanup. This Weld-Location GNN aims to formulate a complex averaging function over the points in the itemized weld cloud that locates the anchor or center location of the weld shape. The Weld-Location network introduces two custom pooling operations that are used side by side: similarity-weighted pooling and proximity-weighted pooling. Both permutation-invariant pooling operations average the messages aggregated from neighboring nodes, but with relative normalization constants that are based on cosine similarities and inverse Euclidean distances, respectively. Additionally, max- and mean-pooling operations are used side by side, along with the weld's centroid in the extraction of the graph-embedding before a final MLP predicts the weld's center location.

2.3.1. Graph generation for weld-location

The weld extractions itemized upstream are re-cast into a graph structure of vertices and edges, $G(V, E)$, where edges exist between any pair of points that are, at most, 1.2 mm apart:

$$E = \{\ (i,j)\ \mid\ \|V_2[j] - V_1[i]\| \le 1.2\ \text{mm}\ \}. \tag{4}$$

2.3.2. Dataset for weld-location

Datasets were generated by saving the post-segmentation clustering extractions as weld point clouds, and their corresponding 2D location labels from the original weld-segmentation label text files. The two clustering schemes (Spatial-Clustering and Segment-Clustering) generate two different styles of weld extractions, which were both used in training the network (with their strengths and faults in tow). The Segment-Clustering dataset had a total of 2773 samples. The Spatial-Clustering dataset had fewer samples of 2657, on account of the merged weld extractions.

2.3.3. Weld-location network architecture

The rationale behind the Weld-Location GNN was to learn a continuous value function to approximate a complex averaging scheme on all the points in the graph. The function should notice the offset relative to the mean position of the weld shape. It should de-emphasize the impact of the vertices that would worsen the prediction accuracy but pay more attention to the high-impact points that hold true to the weld shape. Similarity-Weighted Pooling and Proximity-Weighted Pooling were attempted as alternatives to max-pooling early in the network to keep the node embeddings relatively more differentiated than with max-pooling.

Similarity-Weighted Pooling is defined as:

$$h_u^{(k+1)} = \frac{\underset{v\in N(u)}{\text{SUM}}\left(\dfrac{h_v^{(k)} \cdot h_u^{(k)}}{\|h_v^{(k)}\|\ \|h_u^{(k)}\|} h_v^{(k)} \right)}{\underset{v\in N(u)}{\text{SUM}}\left(\dfrac{h_v^{(k)} \cdot h_u^{(k)}}{\|h_v^{(k)}\|\ \|h_u^{(k)}\|} \right)} + h_u^{(k)}. \tag{5}$$

Proximity-Weighted Pooling is defined as:

$$h_u^{(k+1)} = \frac{\underset{v\in N(u)}{\text{SUM}}\left(\dfrac{1}{\|X_u - X_v\|} h_v^{(k)} \right)}{\underset{v\in N(u)}{\text{SUM}}\left(\dfrac{1}{\|X_u - X_v\|} \right)} + h_u^{(k)}, \tag{6}$$

where $h_u^{(k)}$ and $h_u^{(k+1)}$ represent the previous and updated node embeddings of node u; X_u denotes the 3D coordinate location of node u; $N(u)$ are the neighbors of node u; and $\underset{v\in N(u)}{\text{SUM}}$ is the element-wise sum pooling across neighbors of node u.

The outputs from the respective pooling operations were concatenated together, leaving the model to work out its relative preference between the pooling outputs. Node coordinates were forwarded into later node embeddings, for spatial equivariance and to eliminate the need to center the point cloud at (0, 0). Max-pooling and mean-pooling outputs were concatenated into the graph's initial embedding. The cluster's mean location was forwarded into the graph's initial embedding too, improving the location predictions. The chain of operations of the Weld-Location network is illustrated in Fig. 8.

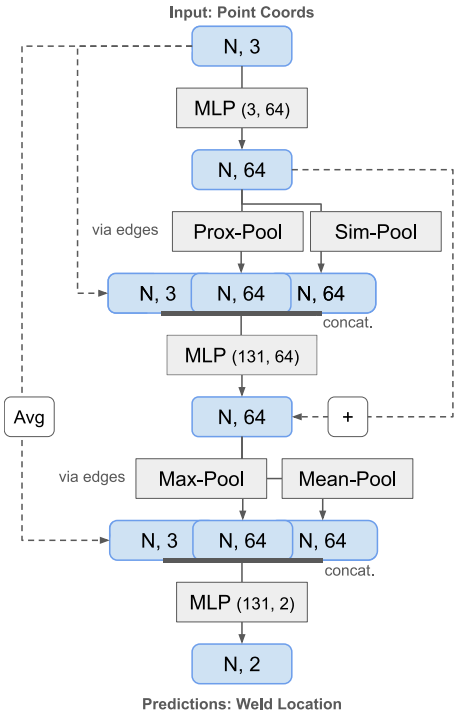


Fig. 8. Information flow of Weld-Location GNN, featuring node-level embeddings pooling into a graph-level embedding of the itemized weld point cloud.

2.3.4. Objective function featuring huber loss

The network was trained with PyTorch's Huber Loss in the objective function. This is a linear function with respect to the error magnitudes, except for below a specified threshold when a quadratic function ensures smoothness. The cost function had a regularization term to limit the magnitudes of the model parameters:

$$\sum_{i=1}^{N} \text{Huber}\left(p_i, t(i) \right) + \sum_{\forall w_k \in W} \|w_k\|, \tag{7}$$

where $t(i)$ is the truth label for the ith sample in the set; p_i is the predicted location for the ith sample; and W is the set of all model parameters.

2.3.5. Training for weld-location

The model was trained in a 5-fold cross-validation experiment. The graphs were, again, constructed with less than the full complement of available edges as a regularization technique. Each Fold was set to train for 2000 epochs with early stopping beyond the 300th epoch. Each model was trained stochastically using PyTorch's stock Adam optimizer with a 0.0001 learning rate. The detections, location predictions, and errors of welds itemized from a sample scan are illustrated in Fig. 9. Fig. 9(d) shows the absolute errors in the predictions for each weld identified, and the latitudinal and longitudinal decomposition of the errors to emphasize the respective primary directions of each error.

3. Results and discussions

3.1. Weld-segmentation GNN results

The first-stage segmentation model was trained and evaluated on 481 and 121 samples, correspondingly. Across the 5 folds, the semantic and instance segmentation subtasks showed average global accuracies of 0.974 and 0.96, and mean Intersection over Union (mIoU) scores of 0.856 and 0.718. In this instance, the accuracy metric reports the

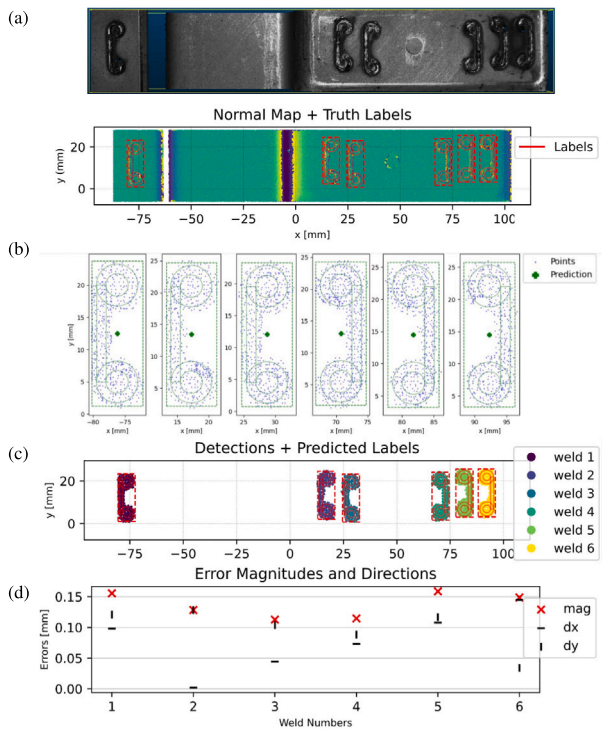


Fig. 9. (a) Gray-scale intensity image (top) and ground truth location labels overlaid on the normal map of the scan (bottom); (b) Weld-Location predictions on each itemized weld, previously isolated from the segmentation predictions of the Weld-Segmentation network; (c) Predictions of weld detections and locations according to the two-stage GNNs aggregated together; and (d) Absolute error and the directionally decomposed error magnitudes (dx and dy) for each weld location prediction compared to the truth labels.

Table 1
Semantic segmentation summary statistics.

Class:	Per weld-direction			Global
	Left-facing	Right-facing	Background	
Accuracy	0.979	0.983	0.974	0.974
[m]IoU	0.793	0.803	0.973	0.856
Recall	0.807	0.813	0.999	0.873
Precision	0.979	0.983	0.974	0.979
F1 Score	0.882	0.888	0.986	0.919

percentage of correctly predicted point-wise classifications, and mIoU is used in its classic sense for multiple-class segmentation.

Tables 1 and 2 show more granular statistics, per weld-direction and per weld-number. The relatively large number of background points skew the global statistics towards more favorable reporting on the respective global metrics. The slight edge in right-facing weld scores can be attributed to the higher number of right-facing welds (1128 compared to 1046 left-facing welds) in the training dataset. Similarly, the number of scans with at least 6 welds is fewer than those with at least 5 welds, which is fewer still than those with at least 4 welds, and so on. Additionally, while the first weld is, in most cases, on the far left of the scan, and the sixth weld, when available, is always on the far right of the scan, there is more variability in the longitudinal (x-dimension) locations of the second to the fifth welds. Combined together, these distribution characteristics might help explain the observed trends in the granular per-instance statistics. Tangentially, semantic segmentation presents a simpler task in this context than instance segmentation, with fewer classes to distribute across. Consequently, the segmentation network performs better on the semantic task.

3.2. Post-segmentation itemizing results

Tables 3–5 report accuracy and precision statistics from the Spatial-, Segment- and Combo-Clustering sequences on weld clusters extracted from the 602 samples of weld-segmentation predictions. These are not directly comparable with the segmentation statistics on account of having discarded many points in the respective clustering schemes. Incidentally, the Segment-Clustering scheme discards more points than Spatial-Clustering, making segment-clustering statistics more sensitive to misclassifications. Combo-Clustering reflects the counts and statistics of Spatial-Clustering for the most part, except in the situations where cluster merges force the scheme to defer to Segment-Clusters. Consequently, the Combo-Clustering statistics show some degree of correcting the misclassifications of the Spatial-Clustering scheme.

3.3. Weld-location GNN results

The second-stage regression model was trained and evaluated on 1927 and 199 itemized weld point clouds, also across 5 folds. Focusing on a dataset created from the upstream segmentation and then post-processed with Segment-Clustering, the location predictions (and thus, the detection predictions) averaged within 0.244 mm of the truth labels, and with a standard deviation of 0.258 mm. Table 6 additionally shows the percentage accuracies across the 199 test samples, referring to the percentage of predictions that are within selected threshold error distances.

Though the GNN-based method captures all the welds in the 3D point clouds, there is a relatively large difference between the predicted weld positions and the actual weld positions for some of the welds. The success of the algorithm is ultimately determined by these location predictions. In this particular study, the precision requirement of the inspection algorithm dictated that the compatibility limits of the prediction accuracies allowed up to approximately 0.75 mm of misalignment. Per the statistics collated from the 121 test samples (Table 7), 92.8% of the correctly detected welds are within 0.75 mm of the truth locations.

3.4. Sources of error

Weld directions are correctly predicted in all test samples, as with the baseline template-matching algorithm. While the GNN-based weld-detector boasts no false detections and no misclassified welds on typical scans, the accuracy of the location predictions on correctly detected welds is invariably worse than that of the baseline. Analyzing the progression from the initial sample scans to the final weld detection reveals a few notable sources of error: (1) noise in the instance segmentation predictions, and (2) inconsistencies when processing irregularly shaped welds through the weld-location network.

A major contributor to location errors is the weld-location network's limited capability to handle poorly shaped weld extractions, leading to reduced accuracy and predictability. The anomalous shapes fall into two broad categories: truncated staple shapes that show up as exaggerated errors in the y-coordinate (Fig. 10); or blobs that bleed beyond the staple shape and disturb location predictions, largely in the x-dimension (Fig. 11). In some cases, the errors in location predictions were more attributable to mistakes in the welding process or ambiguities in the labeling process. For example, a case of imprecisely formed welds is shown in Fig. 12, where some staples were not cleanly welded on, which affects their labeling and hence their evaluation. Conversely, there are also examples of mistakes that are corrected in the detection, as shown in Fig. 13, where a partial weld was correctly located as if full-shaped.

Table 2
Instance segmentation summary statistics.

Instance:	Per weld-number statistics							Global
	1	2	3	4	5	6	Background	
Accuracy	0.987	0.906	0.873	0.834	0.722	0.835	0.97	0.96
[m]IoU	0.778	0.701	0.636	0.619	0.492	0.60	0.97	0.718
Recall	0.784	0.746	0.650	0.696	0.604	0.679	0.999	0.767
Precision	0.987	0.906	0.873	0.833	0.722	0.835	0.97	0.899
F1 Score	0.872	0.816	0.770	0.756	0.654	0.747	0.985	0.823

Table 3
Spatial-clustering summary statistics.

Instance:	Per weld-number statistics						Instance	Semantic
	1	2	3	4	5	6		
Accuracy	0.824	0.847	0.844	0.833	0.817	0.834	0.835	0.8345
mIoU*							0.822	0.843
mPrecision*	0.831	0.834	0.835	0.852	0.846	0.787	0.831**	

Table 4
Segment-clustering summary statistics.

Instance:	Per weld-number statistics						Instance	Semantic
	1	2	3	4	5	6		
Accuracy	0.824	0.847	0.844	0.833	0.817	0.834	0.835	0.8345
mIoU*							0.844	0.837
mPrecision*	0.834	0.851	0.859	0.848	0.847	0.856	0.849**	

Table 5
Combo-clustering summary statistics.

Instance:	Per weld-number statistics						Instance	Semantic
	1	2	3	4	5	6		
Accuracy	0.820	0.841	0.834	0.833	0.826	0.829	0.832	0.8365
mIoU[a]							0.842	0.843
mPrecision[a]	0.832	0.846	0.857	0.849	0.849	0.852	0.848[b]	

[a] Averaged across 602 samples.
[b] Averaged across row.

Table 6
Location prediction summary statistics on datasets generated by Combo-Clustering.

Average error (mm)	0.243
Min error (mm)	0.016
Max error (mm)	2.488
Median (mm)	0.184
Standard deviation (mm)	0.268
Average standard deviation (mm)	0.258
Errors less than:	
– 1 mm	97.39%
– 0.75 mm	95.97%
– 0.5 mm	94.07%
– 0.25 mm	68.24%
– 0.1 mm	19.3%

3.5. Overall weld detection performance

Table 7 shows a summary of the performance of the proposed GNN-based weld detection algorithm on the 121 evaluation point cloud samples. Its performance is compared against a template-matching baseline implementation, which has been susceptible to missing some welds and flagging false positives.

In summary, the GNN-based approach demonstrates strong robustness against erroneous detections, effectively avoiding missed welds and false positives. Its ability to accurately identify all welds relies on the combined effectiveness of the segmentation network and the clustering scheme used for weld itemization. Although the regression network's accuracy in predicting weld locations is somewhat lower compared to the baseline approach, it still performs well enough in most cases: approximately 8% of predictions exceed the 0.75 mm error threshold set by the downstream inspection system.

Table 7
Comparison of weld detections by traditional template-matching method and the proposed GNN-based approach.

	Traditional	GNN-based
Point cloud images		121
Number of welds (Ground Truth)		582
Number of welds detected	583	582
– Welds missed	12	0
– Samples with missed detections	10	0
– Misclassified directions on correct detections	0	0
– False positives	13	0
– Erroneous detections (missed, false positives)	25	0
Average error across welds to detect (mm)	1.19897	
Average error across correct detections (mm)	0.00175	0.2612
Max error (mm)		7.3831
Standard deviation (mm)		0.905
Error > 0.75 mm Across correct detections	0	42 (7.2%)

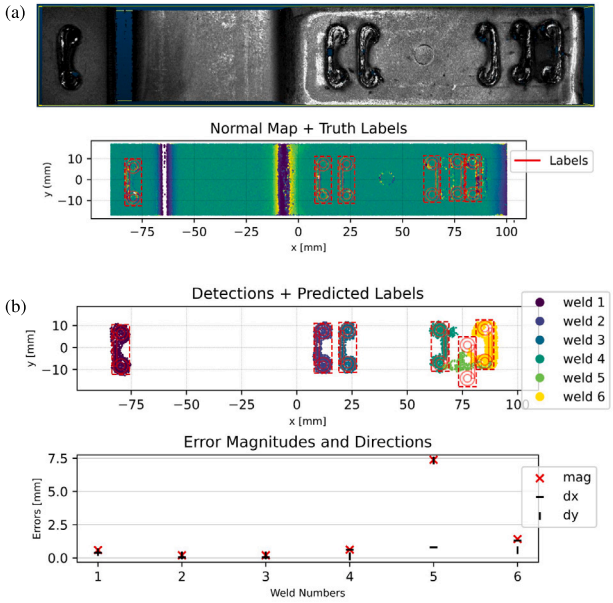


Fig. 10. An example of detecting an incomplete staple-shape weld. (a) The gray-scale intensity image (top) and ground-truth weld locations overlaid on a 2D normal map (bottom) of the scan. (b) The weld-location predictions overlaid on the aggregated plot of post-segmentation weld extractions (top) and the per-weld absolute and decomposed errors in the predicted locations (bottom). The 5th detected weld has a truncated staple shape, showing evidence of deferring to Segment-Clustering. The incomplete shape causes an error in the predicted location of approximately 7.5 mm, which is largely in the y-dimension.

3.6. Computation time

Training and evaluation was processed on an HP Z2 G5 Workstation with Nvidia RTX A6000 (16G) GPU. The proposed GNN-based pipeline takes an average of 0.82 s and a maximum of 0.97 s to predict the weld locations for each point cloud image. This includes generating

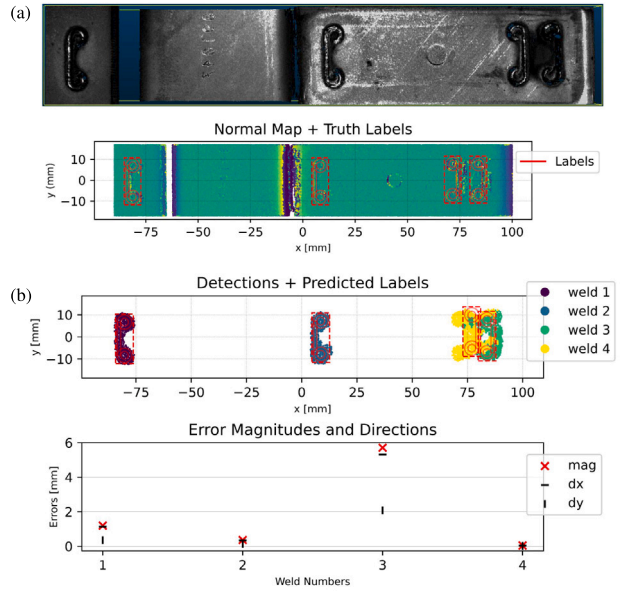


Fig. 11. An example of detecting an ill-formed staple-shape weld. (a) The gray-scale intensity image (top) and ground-truth weld locations overlaid on a 2D normal map (bottom) of the scan. (b) The weld-location predictions overlaid on the aggregated plot of post-segmentation weld extractions (top) and the per-weld absolute and decomposed errors in the predicted locations (bottom). The yellow weld has a bloated shape that encroaches into the green weld, the result of which is a 6 mm error in predicted location, predominantly in the x-dimension.

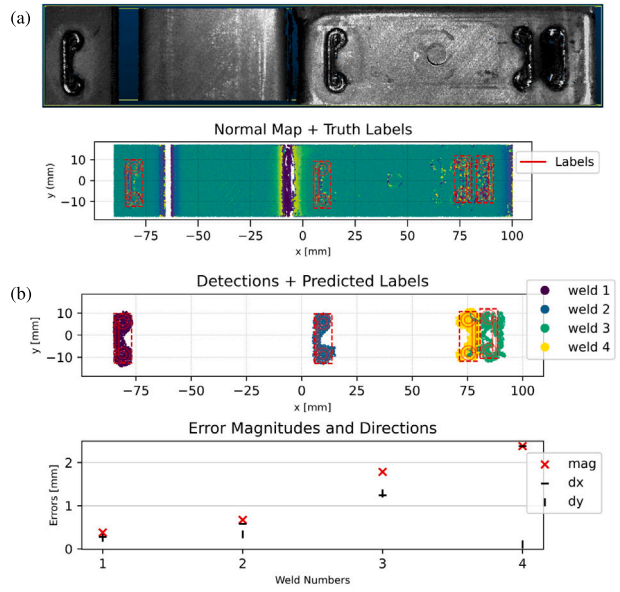


Fig. 12. An example of low-quality labeling. (a) The labeled 2D normal map (bottom) shows coarsely adjusted labels on the 3rd and 4th welds from the left, which are blurry in the normal map indicative of their lower quality. The x-dimension of the labels were particularly challenging. (b) The errors in predictions (bottom) for these welds of roughly 2 mm are hard to defend because of the ambiguous labeling.

output prediction files for the segmentation and location stages of the pipeline, and accompanying plots. By comparison, the baseline alternative executed its weld detection and weld inspection on the 121 test samples in 1.97 s, on average (between 1.517 to 2.821 s), which leaves roughly 1 s within which an inspection-only version can be paired with the GNN-based detector to meet the same rate.

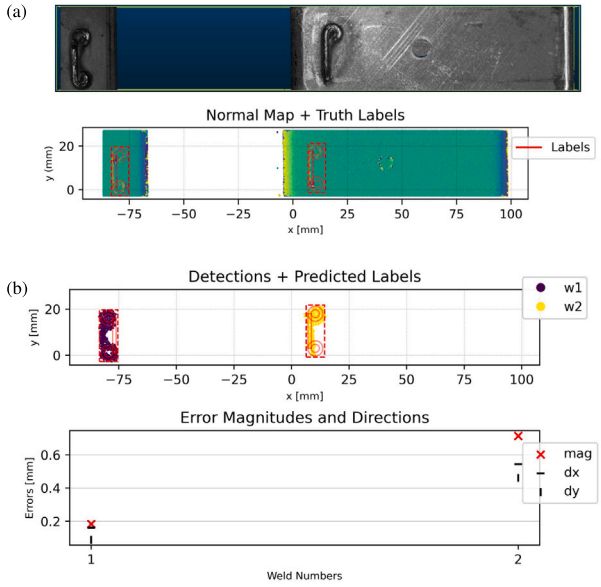


Fig. 13. Location detections on a sample with one partial weld. (a) The gray-scale image shows a clear partial shape of the rightmost weld. (b) The partial weld location is predicted correctly, within the acceptable limits, but the direction of the full-shaped weld is misclassified.

4. Conclusion

The study introduces a graph neural network-based approach for detecting welds in automotive assembly, focusing directly on 3D point cloud data without relying on supplementary inputs like point normals or intensity information. The proposed system effectively mitigates some limitations evident in the baseline template-matching raster-search implementation, which, when confused by the variability in weld appearance and quality, misses some detections and reports some false positives.

A key contribution of this work lies in the chaining of two specialized GNN models: one for segmentation and another for weld localization. This decomposition not only allows for improved detection accuracy but also enhances interpretability, which is particularly desirable for deep learning deployments in industry. This design shows that complex point cloud operations can be broken into simpler, explainable stages without losing effectiveness, contrasting against end-to-end deep learning networks that lack this level of transparency.

The successful validation of this GNN-based pipeline on staple-shaped welds demonstrates its ability to generalize across varying weld conditions and suggests strong potential for broader application to different weld patterns and shapes and in other industrial contexts. The methodology contributes new insights into the application of graph-based learning to unstructured data, and highlights a practical benefit of GNNs in manufacturing environments, namely processing sparse, unstructured, interrelated data effectively.

Having demonstrated the suitability of a two stage GNN network to segment and extract itemized weld clouds, and to locate the weld centers thereafter, there still remains the process of searching the space of hyper-parameters encountered along the way for a globally optimal combination. Key among these hyper-parameters are the DBSCAN parameters, and the counts and sizes of the layers in the respective GNNs. A necessary objective is to increase the model's expressivity, but balanced against its efficiency. Relatedly, re-framing the second stage network as a classification problem instead of regression could improve detection accuracy albeit with relaxed, but acceptable, precision. And finally, we propose to integrate downstream quality inspection directly into the GNN-based framework, creating a full-service and robust detection and inspection system. A short term prospect is to marry the

relative strengths of the GNN-based pipeline and the template-matching algorithm for production use imminently.

CRediT authorship contribution statement

Kojo Welbeck: Formal analysis, Investigation, Methodology, Writing – Original Draft, Writing – Review & Editing, Conceptualization, Data Curation. **Yahui Zhang:** Investigation, Methodology, Writing – Review & Rditing. **Ru Yang:** Conceptualization, Methodology. **Guangze Li:** Funding Acquisition, Resources, Writing – Review & Editing, Conceptualization, Data Curation . **Hui-Ping Wang:** Funding Acquisition, Supervision, Resources. **Blair Carlson:** Funding acquisition, Project Administration, Supervision. **Ping Guo:** Conceptualization, Funding Acquisition, Methodology, Supervision, Writing – Review & Editing.

Declaration of competing interest

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

Acknowledgments

This research was supported by General Motors Company (GM) under Grant Number SU-23-282. The authors would also like to acknowledge support from the National Science Foundation, United States under Award Number EEC-2133630 and CNS-2328032.

References

[1] Mori K, Tarui T, Hasegawa T, Yoshikawa N. Remote laser welding applications for car bodies. Weld Int 2010;24(10):758–63.
[2] Wu D, Zhang P, Yu Z, Gao Y, Zhang H, Chen H, Chen S, Tian Y. Progress and perspectives of in-situ optical monitoring in laser beam welding: Sensing, characterization and modeling. J Manuf Process 2022;75:767–91.
[3] Erdős G, Kemény Z, Kovács A, Váncza J. Planning of remote laser welding processes. Procedia CIRP 2013;7:222–7.
[4] Ceglarek D, Colledani M, Váncza J, Kim D-Y, Marine C, Kogel-Hollacher M, Mistry A, Bolognese L. Rapid deployment of remote laser welding processes in automotive assembly systems. CIRP Ann 2015;64(1):389–94.
[5] Matsunawa A, Mizutani M, Katayama S, Seto N. Porosity formation mechanism and its prevention in laser welding. Weld Int 2003;17(6):431–7.
[6] Xu F, Xu Y, Zhang H, Chen S. Application of sensing technology in intelligent robotic arc welding: A review. J Manuf Process 2022;79:854–80.
[7] Li Y, Hu M, Wang T. Visual inspection of weld surface quality. J Intell Fuzzy Systems 2020;39(4):5075–84.
[8] Czimmermann T, Ciuti G, Milazzo M, Chiurazzi M, Roccella S, Oddo CM, Dario P. Visual-based defect detection and classification approaches for industrial applications—a survey. Sensors 2020;20(5):1459.
[9] Ma G, Yu L, Yuan H, Xiao W, He Y. A vision-based method for lap weld defects monitoring of galvanized steel sheets using convolutional neural network. J Manuf Process 2021;64:130–9.
[10] Guo W, Huang X, Qi B, Ren X, Chen H, Chen X. A novel versatile method for generating machining path directly from point cloud. J Manuf Process 2024;120:15–27.
[11] Hinks T, Carr H, Truong-Hong L, Laefer DF. Point cloud data conversion into solid models via point-based voxelization. J Surv Eng 2013;139(2):72–83.
[12] Lee ET, Fan Z, Sencer B. A new approach to detect surface defects from 3D point cloud data with surface normal gabor filter (SNGF). J Manuf Process 2023;92:196–205.
[13] Nixon MS, Aguado AS. 5—High-level feature extraction: Fixed shape matching. Featur Extr Image Process Comput Vis 2020;223–90.
[14] O'Shea K. An introduction to convolutional neural networks. 2015, arXiv preprint arXiv:1511.08458.
[15] Shi W, Rajkumar R. Point-gnn: Graph neural network for 3d object detection in a point cloud. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 1711–9.
[16] Qi CR, Su H, Mo K, Guibas LJ. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 652–60.
[17] Qi CR, Yi L, Su H, Guibas LJ. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Adv Neural Inf Process Syst 2017;30.
[18] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. Adv Neural Inf Process Syst 2017;30.
[19] Ross T-Y, Dollár G. Focal loss for dense object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 2980–8.
[20] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[21] Kingma DP. Adam: A method for stochastic optimization. 2014, arXiv preprint arXiv:1412.6980.
[22] Deng D. DBSCAN clustering algorithm based on density. In: 2020 7th international forum on electrical engineering and automation. IFEEA, IEEE; 2020, p. 949–53.