Journal of Manufacturing Processes 120 (2024) 867–877

Contents lists available at ScienceDirect

Journal of Manufacturing Processes

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

Forming force prediction in double-sided incremental forming via GNN-based transfer learning

Songlin Duan [a], Dominik Kozjek [a], Edward Mehr [b], Mark Anders [b], Jian Cao [a,*]

[a] Department of Mechanical Engineering, Northwestern University, Evanston, IL 60208, USA [b] Machina Labs, Chatsworth, CA 91311, USA

ARTICLEINFO

Keywords:
Double-sided incremental forming
Forming force prediction
Graph neural networks
Transfer learning

ABSTRACT

This paper proposes a transfer learning approach using graph neural networks (GNN) for predicting the forming force during double-sided incremental forming (DSIF) processes. In order to address the geometry complexity of DSIF parts, a GNN-based model was proposed to aggregate surface geometric information of DSIF parts and toolpaths. Furthermore, a transfer learning method was adopted to improve the prediction. The model was pre- trained on a dataset of previously formed DSIF parts with varying geometries. To address material and machine variations, the model was further trained on the initial few layers of the observed part for calibration and subsequently predicted the forming force in the vertical direction relative to the part's coordinate system for the rest of the layers of the observed part. The performance of our proposed approach was evaluated using experimental datasets from two different machines and different input materials, demonstrating the generality and effectiveness of the approach in forming force prediction.

1. Introduction

The ability of Incremental Sheet Forming (ISF) to manufacture components with asymmetric geometries while reducing tooling costs and development cycle time has made it particularly desirable for prototype development and small-scale production. Double Sided Incremental Forming (DSIF) progressively deforms sheet metal with a tool on both sides to achieve the desired shape [1,2]. Like many flexible manufacturing processes, the ability to achieve dimensional accuracy at the first run is highly desirable and challenging due to the complex process mechanics involved in DSIF. A significant factor affecting dimensional accuracy is springback. To mitigate springback, Ren et al. [3] proposed an in-situ springback compensation method using a model- based feedback control, which can achieve an averaged dimensional accuracy of about 1 *mm*, compared to more than 3 *mm* if no control was implemented. One critical element in achieving such an improvement is the ability to predict instant forming force, which is influenced by several factors, such as tool dimensions [4], sheet metal thickness [5], material types [6], wall angles of target parts [7], lubrication [4], and processing conditions including vertical incremental depth [8], position arrangement between the two tools in DSIF [9], spindle revolution speed [10], and feed rate [11].

Conventional finite element simulations have been widely employed to predict the necessary forming force in incremental forming [12,13]. However, these models have limitations due to large computation resource requirements (from days to weeks) or simplified assumptions such as material linearity and idealized boundary conditions. Thus, data-driven approaches have gained interest for efficient and accurate prediction of forming forces in incremental forming, taking into account the complex interplay of factors within the process. For example, Oraon et al. [14] proposed an artificial neural network (ANN) model to predict the deformation force in incremental sheet forming (ISF) by using AA3003 alloy. The model considered process parameters, sheet thickness, part wall angle, and lubrication conditions. Alsamhan et al. [15] developed a force predictive model for single-point incremental forming (SPIF) using an adaptive neuro-fuzzy inference system (ANFIS), an ANN, and a regression model. The study aimed to determine how the process parameters, tool diameter, and sheet thickness affected the maximal forming force. Liu and Li [16] proposed a backpropagation neural networks (BPNN) model for forming force prediction in the SPIF. To address the limitations of little experimental data, they developed a particle swarm optimization algorithm–based virtual data production strategy in accordance with the mega trend diffusion function. Note that all the above referenced works were for single point incremental forming, which is relatively simpler in mechanics and the complexity of formable geometry compared to DSIF.

Graph neural networks (GNN) have proven effective in modeling complex relationships in structured data, such as graphs, and have been applied in various fields, including material property prediction [17], network-based optimization problems [18], and so on. For example, in the manufacturing field, Mozaffar et al. [19] developed a GNN model to predict thermal responses in the Directed Energy Deposition (DED) process. In their study, the thermal histories of DED for different part geometries were successfully predicted, achieving a geometry-agnostic prediction. In order to reduce computational costs and enhance the training accuracy of neural networks, transfer learning serves as a valuable approach to enable the utilization of pre-trained models in a particular domain to enhance performance in related but distinct domains without requiring a massive amount of data for training. For example, Chattopadhyay et al. [20] developed a transfer learning framework for fatigue stage detection using experimental bio-signals. In their study, the proposed framework significantly improved the fatigue stage classification accuracy.

* Corresponding author.
E-mail address: jcao@northwestern.edu (J. Cao).

https://doi.org/10.1016/j.jmapro.2024.04.093 Received 17 April 2024; Accepted 30 April 2024
Available online 7 May 2024

This paper proposes a GNN-based transfer learning method for forming force prediction in DSIF. The method considers relationships among datapoints in toolpath sequence to take into account more complicated local geometries, as opposed to focusing on individual or non-combined parameters. In this method, a graph structure is applied to the DSIF part and utilizes the structural information contained with surface geometries and toolpath sequences to predict the forming forces measured during the process. Specifically, the model consists of three components: a graph encoder, a graph decoder, and a transfer learning module. The graph encoder converts the geometries of the parts into graph structures, and the graph decoder predicts the target forming force based on the encoded graph structure. Furthermore, the transfer learning module replicates some of the weights from the GNN module and fine-tunes the rest weights to predict the vertical component of forming force based on the learned graph representation. Fig. 1 provides an overview of the proposed model's workflow: The GNN-based force prediction model is pre-trained using a dataset of previously manufactured DSIF parts with varying geometries. Then the initial several layers of the target part are used for further training the model which then predicts the forming force of the remaining layers. The concept of "layer" is derived from the process of forming a cone-shaped part, for example, where the forming tools start by deforming the outermost circle of the cone and progressively move inward, deforming concentric circles with incremental depths. Each of these concentric circles is considered a "layer" of the formed part.

2. Methodology

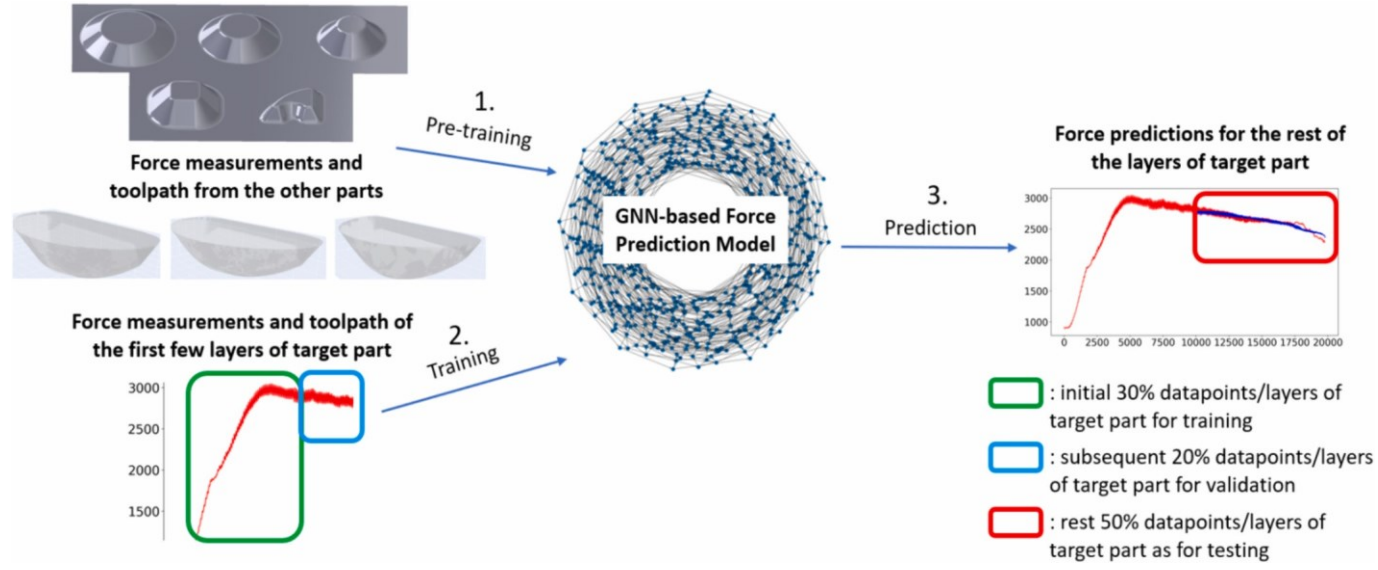

Fig. 1. A flowchart overview of the proposed model. The GNN model was pre-trained on a dataset containing DSIF parts of different geometries and was further trained on the initial few layers of the target part (initial 30 % datapoints as training set and subsequent 20 % as validation set) and then predicted the z-component forming force on the last 50 % datapoints of the target part.

This section describes the methodology used to develop the GNN- based transfer learning approach for forming force prediction in DSIF. In particular, this section details the dataset used in the model, the data preprocessing steps, the model architectures for both GNN and transfer learning methods, as well as the training procedures.

2.1. Dataset

The research employed a dataset consisting of various samples of DSIF process data collected from a real-world manufacturing environment. The datasets are obtained from two different machines and two different materials. Machine 1 applies Computer Numerical Control (CNC) in a gantry system to regulate the two tools to move along the designated paths: a forming tool/upper tool and a supporting tool/ bottom tool, as shown in Fig. 2(a) [21]. The *XYZ* coordinates for each datapoint and its corresponding three-direction components forming force exerted on the forming tool are measured by a force-torque load cell affixed atop the forming tool when the tooltip contacts the metal

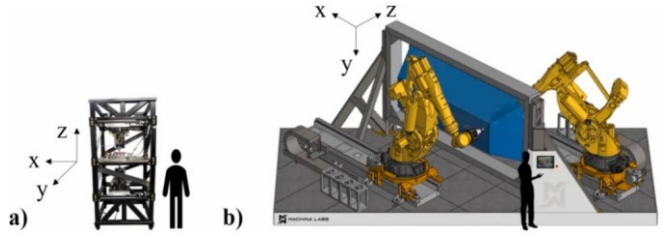

Fig. 2. Configurations for two machines with human-scale. (a) CNC system DSIF machine at Northwestern [21] – machine 1; (b) Robotic manufacturing DSIF system at Machina Labs [22] – machine 2. Coordinates x and y denote the surface plane of metal sheet, while z denotes axial direction of the forming tools.

sheet, denoted as x, y, z, F_x, F_y, and F_z. In this study, F_z serves as the target variable. In each part, x, y in Machine 1 denotes the two- dimension directions along the sheet metal surface which is perpendicular to the axial direction of the tool, while z denotes the axial direction of the tool. The sampling rate of the load cell is set at 20 *Hz* and the linear speed of the tool is 5 *mm/s*. Machine 2 employs an automated robotic manufacturing system applying pressure onto the metal sheet, as shown in Fig. 2(b) [22]. These robots engage in a simultaneous motion of pushing into the sheet along the z direction while deforming the sheet in the *XY*-plane in accordance with a predefined toolpath. The x, y, z coordinates and F_x, F_y, and F_z are gathered by sensors embedded into the robots, whose sampling rate is 1 *Hz* and the tool's linear speed is ~37 *mm/s*. The maximum forming size of collected parts from Machine 1 is from − 55 *mm* to 55 *mm* in the x and y directions and from 0 to − 21 *mm* in the z direction, while in Machine 2, the size is from − 400 *mm* to 400 *mm* in the x direction, from − 200 *mm* to 200 *mm* in the y direction, and from 0 to − 220 *mm* in the z direction.

After capturing the coordinates data, the normal vectors of the part surfaces are computed, denoted as n_x, n_y, and n_z based on the *open3d* package [23] in Python, an open-sourced library designed for handling 3D point cloud data. Furthermore, the curvatures, including both in- plane and vertical curvatures, denoted as $curv_{xy}$, and $curv_z$, are calculated based on every three points with *XYZ* coordinates. The term "in- plane" refers to curves which run along the toolpath and approximately oriented parallel to the *XY*plane. Conversely, "vertical" denotes the curvatures of curves that are nearly

perpendicular to the toolpath directions. The curvature calculation is completed during the pre- processing part. Table 1 lists the parts, and corresponding machine and input material, where number 1 and 2 denote parts from Machine 1 and Machine 2, respectively. The parts from Machine 1 are formed using a spiral line toolpath with an incremental depth of 0.2 *mm*, while the parts from Machine 2 are formed using a contour line toolpath with an incremental depth of 0.3 *mm*. The term "contour" indicates that each datapoint on a given circle (or "layer") possesses the same depth, whereas "spiral" refers to formations where each point exhibits a gradually increasing depth. Fig. 3 shows the shapes of the parts formed by the two machines, where (a)–(e) represent parts from machine 1, and (f)–(h) represent parts from machine 2. To note, parts (f) and (g) appear quite similar in terms of dimensions (length, width, depth). However, there are differences in the wall angles along the left wall lines in the YZ plane. These differences, though minor, significantly impact the forming force, as evidenced by the data we collected in Fig. 4(d). All the eight parts represent distinct geometries and forming force exerted on the forming tool, as measured by load cells. Fig. 4 displays the recorded force curves after a rolling-averaged process. The rolling average will be illustrated in the subsequent Section 2.2.

2.2. Preprocessing

Prior to training the model, several preprocessing steps were performed to ensure the data is suitable for training. First, the outlier points were removed from the dataset by roughly visualizing the data. The outlier points in the dataset were the points when the tools remained stationary. The outliers originated during the period when the tools did not initiate movement, yet the load cell continued to record data points. Subsequently, due to the substantial part size difference and variance in sampling rate between parts from Machine 1 and Machine 2, a re- sampling process was undertaken aiming to ensure a uniform and consistent distance between adjacent data points for all parts. The distance must be selected to capture important trends/variations in forming force and geometry in the data. Meanwhile, the re-sampling rate should not be too high to avoid having too many data points. The latter could lead to more complicated data management and slower learning without improvements in the prediction performance. Based on expert knowledge of the DSIF process specifics, an

Table 1
Dataset.

Part name	DSIF machine	Input material
Cone1_AMPL	1	1 *mm* AA5754-O
Cone2_AMPL	1	1 *mm* AA5754-O
Cone3_AMPL	1	1 *mm* AA5754-O
Pyramid_AMPL	1	1 *mm* AA5754-O
Fish Fin_AMPL	1	1 *mm* AA5754-O
Part1_ML	2	0.9 *mm* 304 L
Part2_ML	2	0.9 *mm* 304 L
Part3_ML	2	0.9 *mm* 304 L

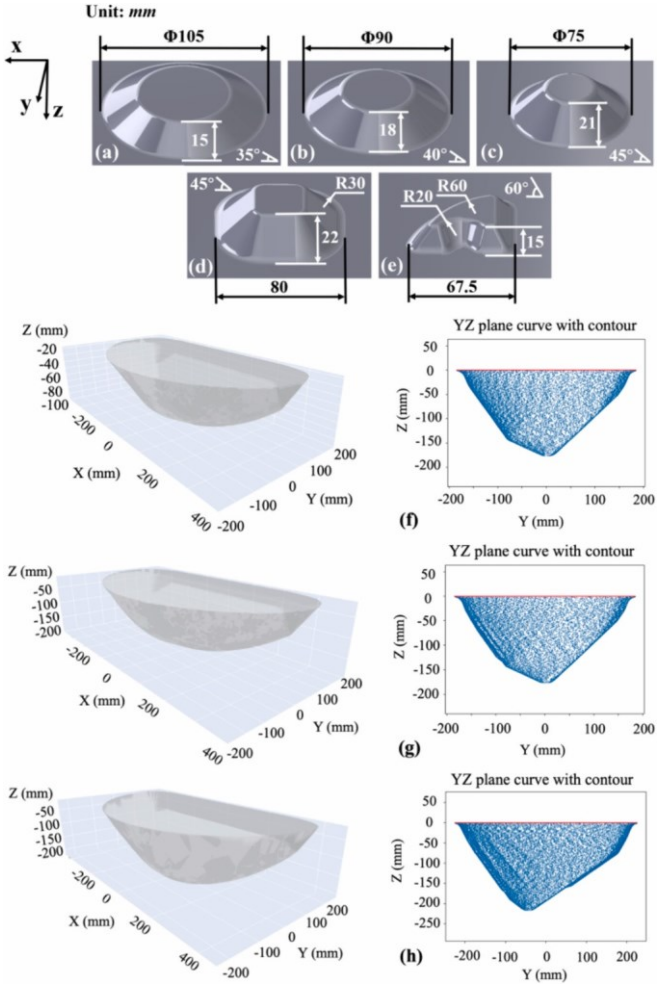

Fig. 3. DSIF parts used in the analysis. (a) Cone1_AMPL, (b) Cone2_AMPL, (c) Cone3_AMPL, (d) Pyramid_AMPL, (e) Fin_AMPL, (f) Part1_ML, (g) Part2_ML, and (h) Part3_ML. (a)–(e) represent parts from machine 1, while (f)–(h) represent parts from machine 2.

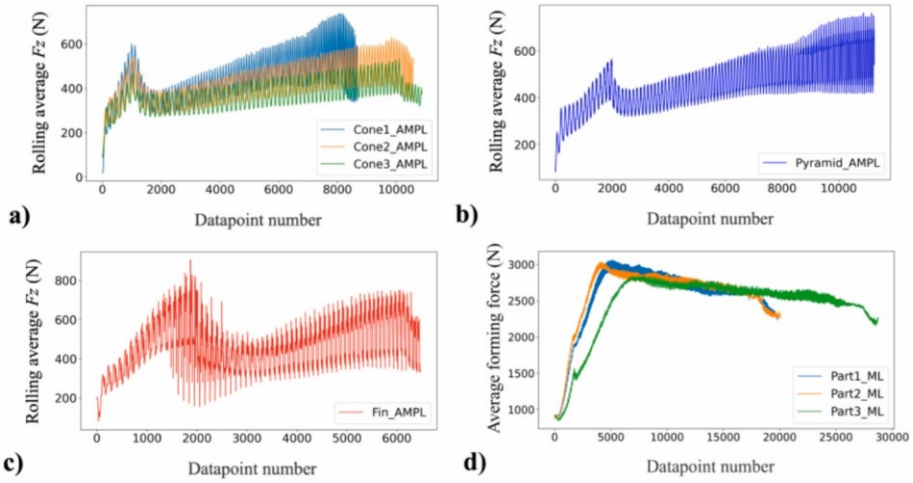

Fig. 4. Force curve illustrations. (a) Force curves of cones from Machine 1, (b) Force curve of pyramid from Machine 1, (c) Force curve of fin from Machine 1 (d) Force curves of parts from Machine 2. Datapoint number means sequential number of a datapoint collected for a part during DSIF forming process.

estimation and decision were made to set the re-sampling adjacent-point distance at approximately 40 *mm* for parts from both Machine 1 and Machine 2 which worked well as later demonstrated by the results. This process serves to mitigate the impact of variations in part size and sampling rates. The total number of data points for all parts before/after re-sampling is 101,000 and 71,750, respectively. Then, the *XYZ* coordinate data were standardized so that the various coordinate features had a reference center with zero means. This step considered the effects of forming symmetries and machine biases, preventing the model from over- emphasizing those effects. Following this, a rolling average was computed by averaging the forming force for a window of 30 data points based on the original/non-resampling data. A window of thirty points is enough to eliminate possibly existing position-dependent measurement errors, and at the same time, it is not too large and still allows within- one-layer predictions. This step aims to prevent learning the measurement uncertainties. However, the downside of applying the rolling average is eliminating the ability to learn and predict local variations in the forming, and therefore, the prediction may not attain an absolute level of local accuracy in this regard. Besides, given that the other parameters, like material properties, process parameters, and metal sheet thickness, of the parts produced by the same machine are similar, the assignment of binary labels, namely 0 and 1, is utilized to differentiate between parts originating from Machine 1 and Machine 2, respectively. The labeled feature is denoted as process parameters shown in Fig. 5.

The curvature of every data point is also needed to provide additional location information when the curvatures are calculated based on the original and non-resampled toolpath points. The calculation logic of curvature is as follows:

Assuming that the three consecutive points (A, B, and C, where B is the middle one) with known *XYZ* coordinates lie on the same curve line, to determine the approximated curvature at the central point B, it is imperative to first obtain the inner circle of the three points. Subsequently, the target curvature value can be obtained by calculating the reciprocal of the radius of the circle. Also, to calculate the coordinates of the inner circle, the basic plane equation forms of three planes, whose intersection point is the circle center, should be calculated first. The three planes are: plane 1 passes through all the three points, while planes 2 and 3 are perpendicular to ~~*AB*~~ and ~~*BC*~~, respectively, and also pass

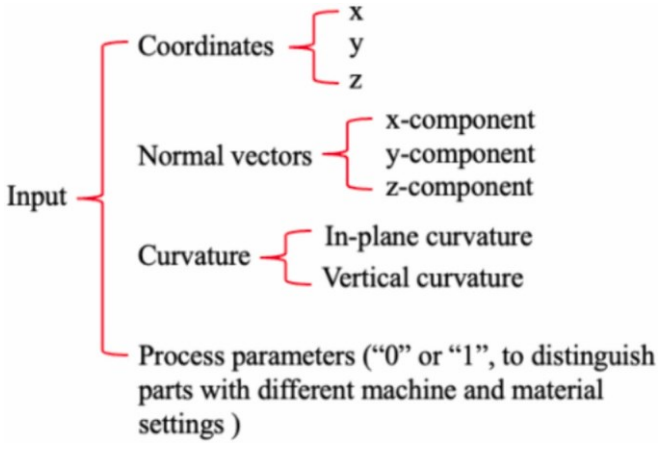

Fig. 5. Input features: coordinates, normal vectors, curvatures, and process parameter.

through the midpoints of ~~*AB*~~ and ~~*BC*~~, respectively. To calculate their basic equation forms, the three planes' normal vectors are needed to calculate first. Following the above produces, the calculations of curvature is as follows:

- The coordinates of the three points are presented below, where B is the middle point:

$$A(x1,y1,z1),B(x2,y2,z2),C(x3,y3,z3) \quad (1)$$

- Calculate the vectors ~~*AB*~~ and ~~*BC*~~ by subtracting the *XYZ* coordinates of A from B and the *XYZ* coordinates of B from C, respectively:

$$\overline{AB} = (x2 - x1, y2 - y1, z2 - z1) \quad (2)$$

$$\overline{BC} = (x3 - x2, y3 - y2, z3 - z2) \quad (3)$$

- Calculate the normal vectors ~~*n1*~~, ~~*n2*~~, and ~~*n3*~~ of the three planes, where ~~*n1*~~ is the normal vector of the plane passes through all the three points, while ~~*n2*~~ and ~~*n3*~~ are normal vectors of planes which are perpendicular to ~~*AB*~~ and ~~*BC*~~, respectively. The two planes perpendicular to ~~*AB*~~ and ~~*BC*~~, respectively, also pass through the midpoints of ~~*AB*~~ and ~~*BC*~~, respectively:

$$\overline{n1} = \overline{AB} \times \overline{BC} \quad (4)$$

$$\overline{n2} = \overline{AB} \quad (5)$$

$$\overline{n3} = \overline{BC} \quad (6)$$

- Calculate the above three planes' constant quantities, d_1, d_2, and d_3, respectively, based on the basic plane equation:

$$\vec{n}_i \bullet \vec{r}_i + d_i = 0, i = 1,2,3 \quad (7)$$ where:

$\vec{n}_i$ = ith plane's normal vector. $\vec{r}_i$ = ith plane's passing point coordinates.
d_i = ith plane's constant quantity

- Calculate the intersection point of the above three planes. The intersection point is the circle center of three-point inner circle:

$$\vec{n}_i \bullet \vec{r}_0 + d_i = 0, i = 1,2,3 \#(8)$$

where:

[$\vec{r}_0$ = coordinates of circle center of three-point inner circle,] $\vec{r}_0 = x_0' y_0' z_0$

- Once obtaining the coordinates of the circle center, the radius of the circle can also be obtained by calculating the Euclidean distance from circle center to any of the three points, A, B, and C. The curvature can be subsequently derived by the equation:

$$Curv = \frac{1}{R_0} \quad (9)$$

where:

Curv = Curvature of the middle point.
R_0 = Radius of the three-point inner circle.

Both in-plane and vertical curvatures for each point are calculated based on the above calculation process. In-plane curvature is calculated by selecting the previous, current, and subsequent point along the toolpath, while vertical curvature is calculated by selecting closest points from previous and subsequent layer/contour.

Fig. 5 outlines the features for approximating the part geometries and other properties. The part geometries are depicted as point clouds, with each point on the cloud representing a point on the surface of the geometry and along the toolpath.

2.3. Graph representations

To incorporate the inherent connections in part geometries and toolpath sequences, a graph representation consisting of nodes and edges was employed to model the point clouds of part geometries and toolpath sequences. In particular, each point in the clouds was treated as a node in the graph, and edges of two distinct types were established between the nodes: Along-Path edges and Space-Layer edges. The Along- Path edges denote that nodes are connected based on the toolpath sequence, while the Space-Layer edges indicate that each node is linked to its *k*-nearest neighbors, the nodes from previous layers/contours. The purpose of Along-Path edges is to encode information about the toolpath sequence while the purpose of the Space-Layer edges is to encode information about the local geometrical features. Here, the parameter *k* determines the number of the closest neighbors considered, and *k* was set to 4 in this study. Various configurations of parameter *k* are tested, and their performances of both calculation time and prediction accuracy are assessed in the subsequent Section 3.4 dedicated to specific target prediction. The Space-Layer edges can solely connect the nodes that are in different layers. Both edges are directed, with the arrow directions from previous nodes towards subsequent ones, which means that the connections are from previous points/nodes. The directed edges ensure that the node is never connected to nodes from subsequent layers, which is expected to consider sequence effect as a hypothesis: when in forming, the in-situ forming force is mostly affected by the formed parts.

Upon constructing the nodes and edges, the weight matrix of neighboring nodes was calculated by GNN, and the aggregated features for each node were updated during training by referring to the features of its neighboring nodes connected through edges. The entire calculation process for weight matrix and aggregated features are depicted from Eqs. (10)–(15). The rationale behind this approach is to encode information about the surrounding geometry and toolpath of each node. Fig. 6 provides an illustrative example of the aggregated calculation.

During the actual forming process, the tool moves along the toolpath direction, from one layer to the next. As shown in Fig. 6, the observed

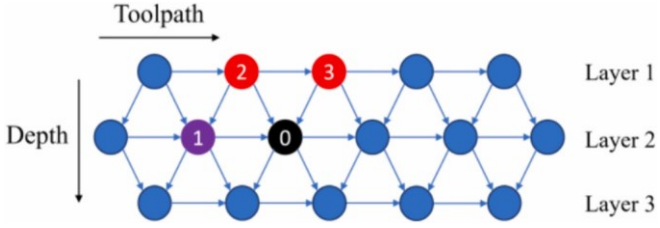

Fig. 6. Detail of the GNN structure. Observed node (black), previous node along the toolpath (purple), and nodes from previous layer (red). Arrows indicate passing the information between the nodes. "layer" refers to each concentric circle formed at increasing depths during forming process. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

node (black node) considers not only its own features but also the features of $Node_1$, $Node_2$, and $Node_3$, as it has an Along-Path edge from $Node_1$ represented as purple arrow in Fig. 6, and Space-Layer edges from $Node_2$ and $Node_3$ represented as red arrows in Fig. 6. The aggregation calculation can be expressed as follows:

- To calculate the aggregated feature vector at $Node_0$:

$$X'_0 = \sum_{i=0}^{3} (M_i \cdot X_i) \quad (10)$$

where: X'_i = Aggregated feature vector of $Node_i$. M_i = Weight matrix of $Node_i$ calculated in GNN X_i = Feature vector of $Node_i$

- To represent the feature vector associated with each node:

Prediction results for all three modes.

$$X_0 = [x_0, y_0, z_0, v_{x0}, v_{y0}, v_{z0}, curv_{xy0}, curv_{z0}, p_0] \quad (11)$$

$$X_1 = [x_1, y_1, z_1, v_{x1}, v_{y1}, v_{z1}, curv_{xy1}, curv_{z1}, p_1] \quad (12)$$

$$X_2 = [x_2, y_2, z_2, v_{x2}, v_{y2}, v_{z2}, curv_{xy2}, curv_{z2}, p_2] \quad (13)$$

$$X_3 = [x_3, y_3, z_3, v_{x3}, v_{y3}, v_{z3}, curv_{xy3}, curv_{z3}, p_3] \quad (14)$$

where: x_i, y_i, z_i = XYZ coordinates of $Node_i$ v_{xi}, v_{yi}, v_{zi} = XYZ coordinates normal vectors of $Node_i$ $curv_{xyi}$ = In-plane curvature of $Node_i$ $curv_{zi}$ = Vertical curvature of $Node_i$ p_i = Process parameter of $Node_i$

- After updating the aggregated feature vectors for each node after training, the prediction function of the forming force can be accomplished using the following equation:

$$F_{zi} = f(x'_i, y'_i, z'_i, v'_{xi}, v'_{yi}, v'_{zi}, curv'_{xyi}, curv'_{zi}, p'_i) \quad (15)$$

where:

F_{zi} = z-component force at $Node_i$ x'_i, y'_i, z'_i = Aggregated coordinates of $Node_i$ $v'_{xi}, v'_{yi}, v'_{zi}$ = Aggregated normal vectors of $Node_i$ $curv'_{xyi}, curv'_{zi}$ = Aggregated in-plane or vertical curvature of $Node_i$ p'_i = Aggregated process parameter of $Node_i$ $f()$ = Relation function between aggregated feature vector and predicted force

The model described above is referred to as the Full-Layer Graph Neural Network, as depicted in Fig. 7.

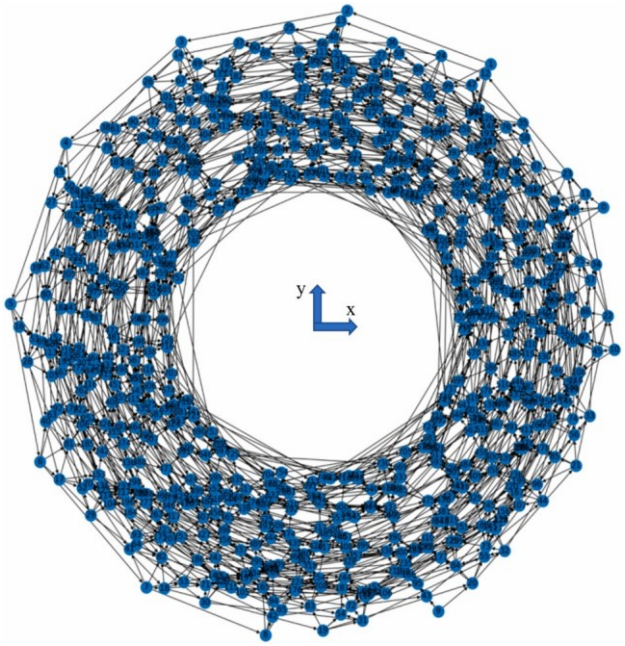

Fig. 7. Example: Full-Layer GNN Model for Cone Structure. Blue nodes are datapoints collected during forming. The sequential numbers marked on nodes denote sequences of datapoints in the toolpath. Black arrows denote edges which pass information between nodes. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2.4. GNN-based transfer learning

A transfer learning method was implemented based on GNN to establish the relationship between the graph representation of part geometries and the corresponding forming forces. The model leverages a GNN architecture with two graph layers (9 × 16 and 16 × 16) and three fully connected layers (16 × 16,

16 × 8 and 8 × 1), with *Tanh* activation function for hidden layers. To note, other activation functions, such as *sigmoid* and *ReLU*, were explored but failed to outperform the *Tanh* function in terms of model performance. The Graph Attention Network (GAT) [24] layers are used as graph layers, which contribute to considering separate importance to every neighboring node. By taking the graph representation of part geometries as input, the GNN learns a latent representation that encodes the structural information of the graph. The resulting latent representation is then transformed into the corresponding forming forces by the output layer. The GNN model was implemented by *PyTorch Geometric* library [25].

To perform transfer learning, the GNN model was initially pre- trained on the pre-training set, which consisted of a subset of training parts with geometries distinct from the target part. Then, the weights of graph layers and the first fully connected layer were frozen in the GNN model, and the weights of the remaining layers were fine-tuned on the training set. This strategy enabled the model to leverage the knowledge learned from the pre-trained model to enhance the performance of the target part with limited training data.

2.5. Training procedures

The datasets were split into different sets, including pre-training, pre-validation, training, validation, and test sets. The pre-trained model uses pre-training and pre-validation, while the transfer model uses training and validation. The two models are evaluated on the test set separately after training or fine-tuning to distinguish the performance difference between models with or without transfer learning. The specific chosen datasets differ based on the pre-trained mode selections and predicted targets. The pre-training set is utilized for pre-training the model, and the pre-validation set fine-tunes pre-trained model's hyperparameters (number of epochs). The training set is used for training on the initial several layers of the target part in the transfer model, while the validation set selects the best number of epochs to end the transfer-learning process. Finally, the test set evaluates the model's performance and compares the prediction differences between the proposed model and the benchmark, and models with and without transfer learning. The split proportions are the random 70 % and the remaining 30 *%* datapoints of individual parts, respectively, for pre-training and pre-validation sets in the pre-trained datasets, and the initial 30 *%,* the subsequent 20 *%*, and the remaining 50 *%* datapoints of the observed part, respectively, for the training, validation, and test sets. It means that, for example, part1_ML is the prediction target in mode 3: pre-training the model with the other 7 parts. After that, the pre-trained model is further trained on the initial 50 % (30 % for training and 20 % for validation to determine the number of epochs) datapoints/layers of part1_ML for calibration. Finally, the model predicts on the test set: the remaining 50 % datapoints/layers of part1_ML.

To provide a benchmark for comparison purposes, a simple ANN model with five fully connected layers (9 × 16, 16 × 16, 16 × 16, 16 × 8, and 8 × 1) was trained and evaluated on the same dataset. The ANN model was employed with the same input features as the GNN model, i.
e., *XYZ* coordinates, *XYZ* component normal vectors, curvatures, and process parameters, but did not incorporate the structural information in the graph which can be only done by the GNN model through connections between adjacent nodes. For the purposes of comparability with the proposed GNN model, the benchmark model was employed a transfer learning approach by freezing the weights of the first three layers and fine-tuning the last two fully connected layers.

Additionally, to contrast the performance of the pre-trained models trained on different datasets, we conducted three pre-training modes. Mode 1 involved pre-training the model solely on parts from Machine 1 and subsequently making predictions for parts from Machine 2. Mode 2 utilized a pre-trained model consisting of only parts from Machine 2, while Mode 3's pre-trained model includes parts both from Machine 1 and Machine 2. The selected pre-training modes are presented in Table 2. In each mode, a comparison was conducted between the performance with and without transfer learning to evaluate the importance of transfer learning. In all three modes (Mode 1, Mode 2, and Mode 3), the trained forming force prediction model is tested on the same data. This enables fair prediction performance comparison between the three modes.

To train the GNN-based model, the mean squared error (MSE) loss function was employed, and the Adam optimizer [26] and learning rate of 0.001 was used, while the learning rate was set to 0.0001 when fine- tuning the not-frozen layers in transfer model. To mitigate overfitting in Mode 1 and Mode 2, an early stopping strategy was implemented with a patience of 10 epochs, and the best training epoch number was selected based on the performance on the validation set. Additionally, in order to avoid stopping at a local minimum in Mode 2, a preliminary training is conducted to estimate the loss changes before confirming the epoch number.

Moreover, the performance between the proposed model and the benchmark was evaluated, as well as between the models with or without transfer learning, and between diverse pre-training mode selections, using the following metrics on the test set:

- Mean Squared Error (MSE): measures the average squared difference between the predicted and actual forming force values and sets as loss function to evaluate the model's training capability.
- Coefficient of determination (R^2 score): compares matching of two sequences of numeric values and measures the proportion of variance in the forming force that is explained by the model. The score range is no more than 1: when the score equals 1, it indicates a perfect match of the two sequences of values; an R^2 score of 0 means

Table 2
Pre-training mode selection.

Mode 1		Mode 2		Mode 3		
Pre-train	Train & test[a]	Pre-train	Train & test[a]	Pre-train		Train & test[a]
Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part1_ML	Part2_ML Part3_ML	Part1_ML	Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part2_ML Part3_ML	Part1_ML
Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part2_ML	Part1_ML Part3_ML	Part2_ML	Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part1_ML Part3_ML	Part2_ML
Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part3_ML	Part1_ML Part2_ML	Part3_ML	Cone1_AMPL Cone2_AMPL Cone3_AMPL Pyramid _AMPL Fin_AMPL	Part1_ML Part2_ML	Part3_ML

[a] The model is further trained on the first few layers of the target part and then tested by making forming force predictions for the rest of the layers of the target part.
Not the same data are used for training and testing the model.

that the model's prediction is no better than simply using the average value of the target values (baseline model); and when the prediction is less accurate than the baseline, the R^2 score is negative.

- Mean prediction error: calculates the average error between predicted and actual forming force values.
- Max prediction error: identifies the highest error between predicted and actual forming force values.

3. Results

3.1. GNN-based transfer learning

Table 3 presents the performance comparison between the GNN- based transfer learning model using Mode 3 and the simple ANN benchmark model. The models are evaluated based on their averaged R^2 scores, averaged mean prediction errors, and averaged max prediction errors for all target predictions. The outcomes demonstrate that the GNN-based transfer learning model using Mode 3 surpasses the simple ANN benchmark model in terms of its overall performance.

3.2. Pre-training mode selection comparison

Table 4 presents the prediction results for all three pre-training modes. In Table 4, "Without TL" denotes the model after pre-training directly predicts on the test set (the final 50 % datapoints of target part), while "With TL" denotes the pre-trained model predicts on the test set after training on training set (initial 30 % datapoints of target part) and validating on validation set (middle 20 % datapoints). It has been demonstrated that pre-training on a dataset that includes both Machine 1 and 2 achieves the best performance based on R^2 scores, mean prediction error, and max prediction error. Additionally, the number of epochs of pre-training for Mode 3 is considerably lower than that of Mode 2, as indicated by Fig. 8. In particular, the loss of Mode 2

Table 3

frequently reaches a local minimum, which requires training iterations to achieve the optimal minimum. Furthermore, in Mode 2, the predicted R^2 score could not be improved significantly by transfer learning approach due to its tendency to overfit. However, the R^2 score could be considerably enhanced when the dataset contains parts from multiple sources in Mode 3.

Fig. 9 depicts the performance evaluation of the best predictive model using Mode 3 by comparing the predicted values with the actual values for Part1_ML, Part2_ML, and Part3_ML, respectively. The obtained R^2 scores for the three target parts are 0.882, 0.883, and 0.809, respectively, indicating an acceptable level of prediction accuracy. Additionally, Fig. 10 displays the prediction results using Mode 3 for the entire window size including training, validation, and testing set, which shows a comprehensive prediction performance of the model, while Fig. 9 shows the prediction performance only in test set which can be seen as a local window size. The results suggest that the predictive model can predict vertical forming force on a layer level.

Moreover, it is essential to investigate the reasons why pre-training solely on one machine may reach a local minimum and require longer training times to find the optimal minimum. A possible explanation is that using a more diverse dataset, incorporating various machine settings and part designs, could help to avoid overfitting to the specific characteristics of one machine and enhance the model's generalization capability.

3.3. With and without transfer learning comparison

In Fig. 11, the comparison results for prediction accuracy, as measured by R^2 scores, mean prediction errors, and max prediction errors, are displayed for the GNN models with and without transfer learning via Mode 3. The results indicate that the GNN model with transfer learning outperforms the one without, as evidenced by an improvement in R^2 scores. Also, there is a noticeable decrease in both

Prediction results for the GNN-based transfer learning model using mode 3 and the ANN benchmark model. The R2 scores, mean prediction errors, and max prediction errors for each model are averaged based on prediction results for all three targets in mode 3.

Models	Averaged R^2 Scores	Averaged mean prediction error (*N*)	Averaged max prediction error (*N*)
GNN-based model	**0.858**	**35.92**	**146.70**
ANN-based benchmark	0.713	51.74	264.58

A bold number indicates a better performance between the two models.

Table 4

Prediction results for all three modes.

Target	Criteria	Mode 1		Mode 2		Mode 3 (BEST)	
		Without TL	With TL	Without TL	With TL	Without TL	With TL
Part1_ML	R^2 score	− 303.47	− 3.18	0.871	0.874	0.786	**0.882**
	Training Time Cost (*s*)	195	1.253	459	**0.078**	118	3.431
	Mean prediction error (*N*)	2178.32	199.26	37.50	37.09	61.16	32.01
	Max prediction error (*N*)	2459.03	648.32	120.87	**119.99**	259.44	121.51
Part2_ML	R^2 score	− 222.17	− 10.63	0.876	0.877	0.843	**0.883**
	Training Time Cost (*s*)	196	1.893	555	**0.047**	253	3.014
	Mean prediction error (*N*)	2189.15	354.85	37.73	37.57	53.43	**37.54**
	Max prediction error (*N*)	2463.42	1820.57	149.39	149.82	**123.84**	151.34
Part3_ML	R^2 score	− 342.28	− 9.81	0.499	0.500	0.129	**0.809**
	Training Time Cost (*s*)	198	5.872	671.06	**0.055**	161.83	12.028
	Mean prediction error (*N*)	2051.19	284.88	66.90	66.84	90.77	**38.20**
	Max prediction error (*N*)	2330.87	999.31	**164.95**	165.35	252.44	167.26

A bold number indicates the best performance of the quantity among all 6 models.

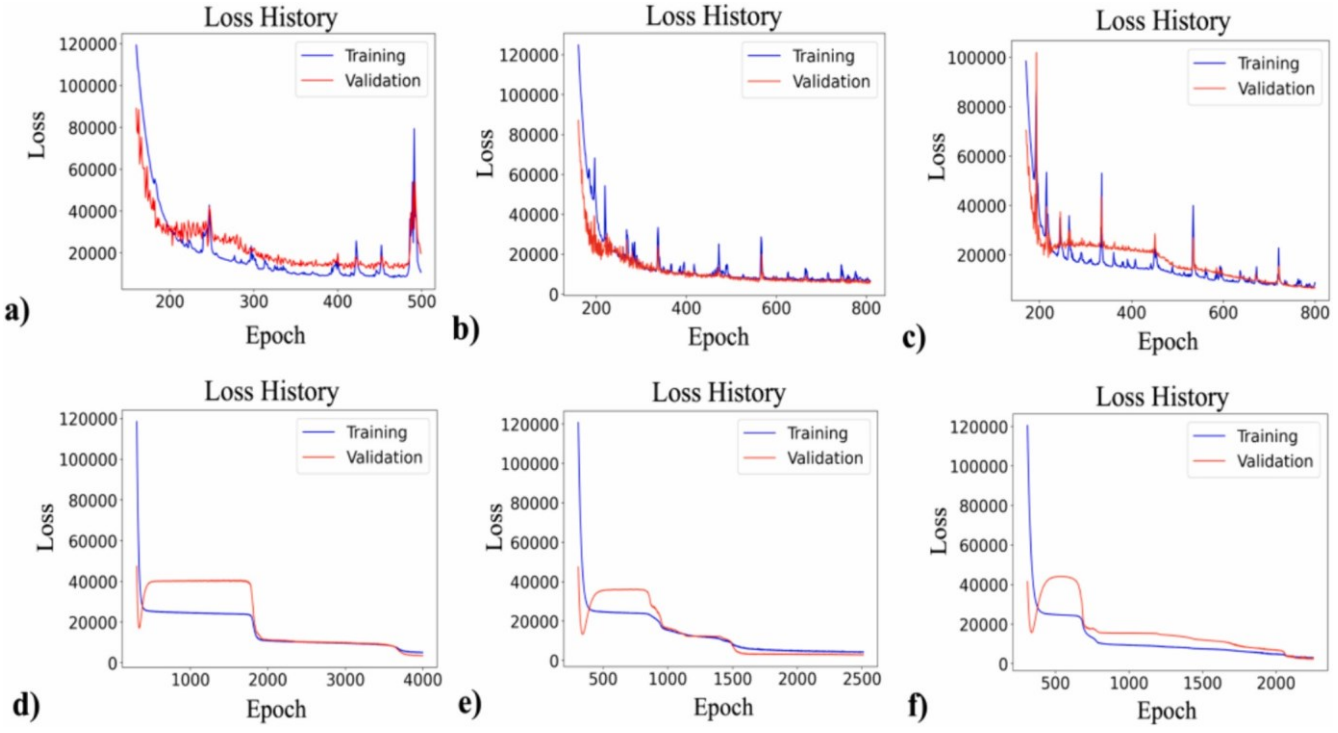

Fig. 8. Training and validation loss history plots for Mode 2 and Mode 3. (a) Part1_ML for Mode 3 (b) Part2_ML for Mode 3 (c) Part3_ML for Mode 3 (d) Part1_ML for Mode 2 (e) Part2_ML for Mode 2 (f) Part3_ML for Mode 2.

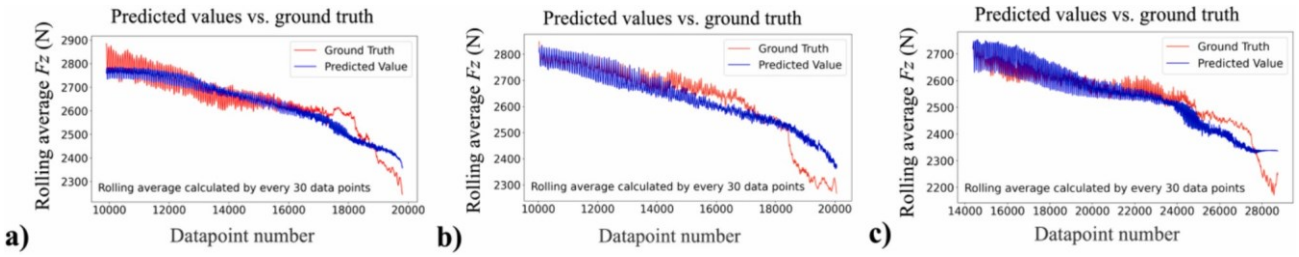

Fig. 9. Prediction results for parts from Machina Labs. (a) Part1_ML (b) Part2_ML (c) Part3_ML. Mode 3 model is used to generate the forming force predictions shown in the figure.

mean prediction error and max prediction error, except for Part2_ML *3.4. Parameter k selection for nearest neighbors* where the max prediction error of the model with transfer learning is

slightly lower than that of the model without transfer learning. To determine the optimal parameter *k* in *k*-nearest neighbors within the graph representation, an array of *k* values was evaluated for

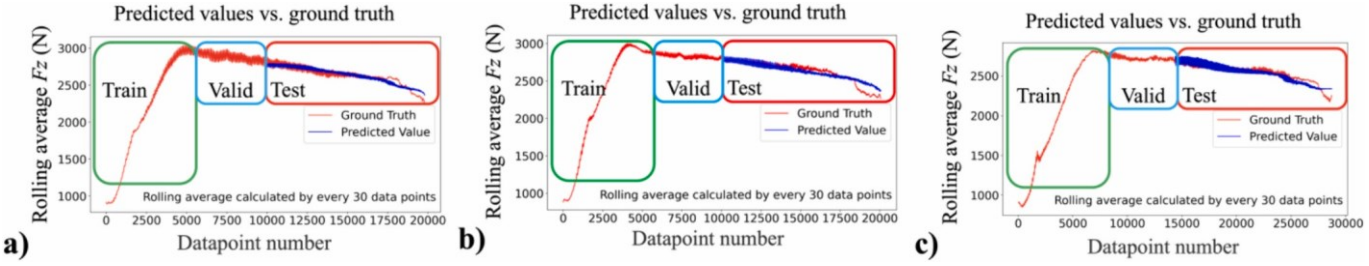

Fig. 10. Actual and predicted forming forces via Mode 3 for target parts. (a) Part1_ML (b) Part2_ML (c) Part3_ML. Green blocks denote the first 30 % datapoints of target parts as training set. Blue blocks denote the subsequent 20 % datapoints of target parts as validation sets. Red blocks denote the last 50 % datapoints of target parts as test set. The models are pre-trained on the pre-training sets. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

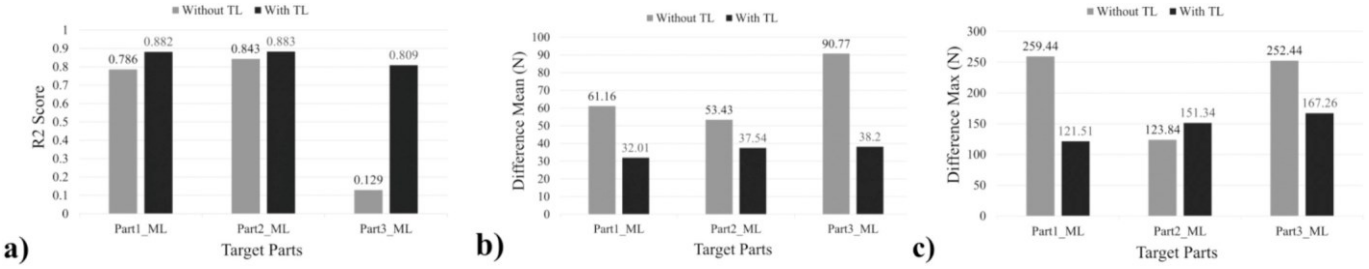

Fig. 11. Comparison results for prediction accuracy with and without transfer learning method. (a) R^2 Scores comparison (b) Mean prediction error comparison (c) Max prediction error comparison. The abbreviation *TL* denotes transfer learning.

Table 5
Part1_ML Prediction results for all parameter *k* selections.

Criteria	k = 2		k = 4		k = 6		k = 8		k = 10	
	Without TL	With TL	Without TL	With TL	Without TL	With TL	Without TL	With TL	Without TL	With TL
R^2 scores	0.661	0.771	0.786	0.882	0.916	**0.920**	0.794	0.873	0.878	0.895
Training Time cost (s)	103	3.226	118	3.431	202	**1.150**	264	4.244	285	1.501
Mean prediction error (N)	60.10	199.26	61.16	32.01	28.82	**28.37**	46.31	33.71	34.84	32.57
Max prediction error (N)	189.79	648.32	259.44	**121.51**	130.52	136.73	182.61	192.81	143.42	126.93

A bold number indicates the best performance among all the models examined in that row.

Part1_ML prediction with pre-training data consisting of parts from both Machine 1 and Machine 2. Table 5 presents the performance trends associated with various *k* values, ranging from 2 to 10. To note, as *k* increases from 2 to 6, there is observable enhancement in performance, evidenced by improved R^2 scores and reductions in both mean prediction error and max prediction error, even with longer training time cost; however, when *k* is further increased from 6 to 8, the performance gains become marginal, and the prediction accuracy decreases compared to when *k* is set at 6. One possible explanation for this phenomenon may be that for this certain task, the Space-Layer edges tend to capture more features as *k* increases and reach their peak performance when k equals 6. Beyond this point, where *k* exceeds 6, the collected features by Space- Layer edges, might introduce disruptive information into the prediction process, leading to a decrease in prediction accuracy.

4. Summary and discussions

In this work, a transfer learning approach based on Graph Neural Network (GNN) for prediction of vertical component of forming force to consider part geometries and the toolpath sequence in the Double-Sided Incremental Forming (DSIF) process is proposed. The study employs a dataset consisting of various experimental samples of DSIF data collected from two different machines and using two different input materials. The results show that the GNN-based transfer learning approach exhibits superior performance

compared to the simple Artificial Neural Network (ANN) benchmark model and the GNN model that does not incorporate transfer learning, as evidenced by the R^2 scores, mean prediction errors, and max prediction errors. Furthermore, the utilization of transfer learning, where the model is trained on a larger dataset and then fine-tuned on a smaller dataset specific to the problem at hand, may also improve the prediction model's performance. Additionally, the results demonstrate that training mode, which involves pre-training with parts from both machines and of different materials, outperforms the other modes in terms of prediction accuracy and pre- training efficiency.

One major achievement of this work is that even with limited dataset (two machines, two material types, and 8 geometries), we have demonstrated noticeable successes in the capability of force prediction. One essential part of this success in data-driven approach is to use forces from initial few layers to calibrate the model. This approach is rooted in the solid understanding of the physics in the DSIF process, as the forming force is a collection of material type, material thickness, machine stiffnesses, etc.

The proposed approach can be material/machine agnostic since we used the initial layers of the target part for training. Any change in material type, material thickness, machine settings, and process parameters will affect the forming force at the initial part of the forming force, therefore, these changes are implicitly incorporated. However, when the dataset is expanded to include more materials, machine settings, and process parameters, the presented approach can be modified by explicitly incorporating some key parameters as separate input features, such as yield strength, material thickness, tool diameter, and tool alignment between the two tools with respect to the sheet surface. This additional information would probably enhance the robustness of the model. Additionally, the effect of the clamping system on forming force has two contributions, locally and globally. Locally, it depends on how effective the clamp system is in terms of controlling metal draw-in. If both clamp systems have the same effect, such as fully preventing sheet metal from slipping under the clamp, then there should not be any effect on forming force. Globally, sheet metal acts as a spring or beam between the clamping system and the forming tool. Therefore, the distance between a forming tool and the clamp edge will influence the forming force. This bias caused by clamp systems in different machines can be captured by the proposed GNN model during training on the initial several layers of the target parts. However, one can also add an extra input feature to the presented model here by considering distances from the tool's contact points on the metal sheet to the four clamps secured at the sheet's four edges during forming. This feature would probably further improve the model's robustness.

Another future research direction can involve the optimization and improvement of the hyperparameters of the model and alternative neural network structures, i.e., time-series algorithms. Furthermore, it is worthwhile to note that the prediction performance for mode 1 with TL is significantly lower, as compared to modes 2 and 3 in Table 4. This may stem from the model's lack of prior knowledge of material or machine characteristics, which are usually learned during the pre-training process. However, based on mode 3 where the prediction results are the best among all three modes and similar materials, shapes, and machine settings are considered in the pre-training dataset, the expectation would be that, by including features of material properties, part shapes, and machine settings similar to the target part in the pre-training dataset, the model could capture the range of material/shape/machine bias in target parts and successfully predict their rest layers with TL. Based on the results, it is recommended to employ a wider range of parts fabricated using various machine settings in the pre-training process. Besides, it could be beneficial to consider not only the z-component of the forming force but also the forces in the xy-plane, as both dimensions of force could undergo significant changes in scenarios where wall angles are either sufficiently high or experience abrupt alterations. Additionally, the target output currently predicted through averaging to mitigate measurement uncertainties may exhibit slight deviations from the actual vertical component forming forces. Moreover, it is essential to consider a tradeoff between re-sampling rate/time required for training and maintaining the integrity of feature resolution. The present prediction model, although provides satisfactory results at the layer level, due to re- sampling and rolling average to mitigate the training time and systematic errors, cannot predict very localized variations. The current selection of distance between points along the toolpath may not necessarily be the optimal one, and to enhance optimization, it is imperative to take into account factors such as tool moving speed, or normalized local features with respect to part size. Also, data imbalance, i.e., the difference in the numbers of datapoints of smaller and larger parts in the training datasets may cause a model to focus too much on large parts. In future work and when using the proposed method, we suggest paying attention to potential data imbalance issues, especially if in a training dataset are data from parts with significantly different sizes. Techniques such as under- or oversampling might need to be utilized to avoid the negative effects of the data imbalance.

Finally, the training time using the transfer learning model (Table 4) demonstrates a great potential for implementing the predictive model for in-situ springback control to achieve the goal of forming the first part right, and therefore, make this process truly autonomous and can be adopted for point-of-need manufacturing or distributed manufacturing.

CRediT authorship contribution statement

Songlin Duan: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Dominik Kozjek:** Conceptualization, Formal analysis, Investigation, Methodology, Validation, Visualization, Writing – original draft, Writing – review & editing. **Edward Mehr:** Conceptualization, Resources, Writing – review & editing. **Mark Anders:** Conceptualization, Resources, Writing – review & editing. **Jian Cao:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Writing – review & editing.

Declaration of competing interest

Authors declare no competing interest in the work of submitted manuscript.

Acknowledgements

We are grateful for the project supported by the Office of Naval Research (N00014-21-1-2484 P00001) and the NSF Engineering Research Center grant (EEC-2133630) "Hybrid Autonomous Manufacturing Moving from Evolution to Revolution (ERCHAMMER)."

References

[1] Meier H, Smukala V, Dewald O, Zhang J. Two point incremental forming with two moving forming tools. Key Engineering Materials 2007;344:599–605. https://doi.org/10.4028/www.scientific.net/KEM.344.599.

[2] Cao J, Huang Y, Reddy NV, Malhotra R, Wang Y. Incremental sheet metal forming: Advances and challenges. Proceedings of the International Conference on Technology of Plasticity, ICTP2008 2008:1967–82.

[3] Ren H, Xie J, Liao S, Leem D, Ehmann K, Cao J. In-situ springback compensation in incremental sheet forming. CIRP Annals 2019;68:317–20. https://doi.org/10.1016/j.cirp.2019.04.042.

[4] Duflou J, Tunçkol Y, Szekeres A, Vanherck P. Experimental study on force measurements for single point incremental forming. J Mater Process Technol 2007; 189:65–72. https://doi.org/10.1016/j.jmatprotec.2007.01.005.

[5] Li Y, Daniel WJT, Liu Z, Lu H, Meehan PA. Deformation mechanics and efficient force prediction in single point incremental forming. J Mater Process Technol 2015;221:100–11. https://doi.org/10.1016/j.jmatprotec.2015.02.009.

[6] Torsakul S, Kuptasthien N. Effects of three parameters on forming force of the single point incremental forming process. J Mech Sci Technol 2019;33:2817–23. https://doi.org/10.1007/s12206-019-0528-2.

[7] Bansal A, Lingam R, Yadav SK, Venkata Reddy N. Prediction of forming forces in single point incremental forming. Journal of Manufacturing Processes 2017;28: 486–93. https://doi.org/10.1016/j.jmapro.2017.04.016.

[8] Barnwal VK, Chakrabarty S, Tewari A, Narasimhan K, Mishra SK. Influence of single-point incremental force process parameters on forming characteristics and microstructure

evolution of AA-6061 alloy sheet. J of Materi Eng and Perform 2019;28:7141–54. https://doi.org/10.1007/s11665-019-04446-9.
[9] Ndip-Agbor E, Smith J, Ren H, Jiang Z, Xu J, Moser N, et al. Optimization of relative tool position in accumulative double sided incremental forming using finite element analysis and model bias correction. Int J Mater Form 2016;9:371–82. https://doi.org/10.1007/s12289-014-1209-4.
[10] Xu D, Wu W, Malhotra R, Chen J, Lu B, Cao J. Mechanism investigation for the influence of tool rotation and laser surface texturing (LST) on formability in single point incremental forming. Int J Mach Tool Manuf 2013;73:37–46. https://doi.org/10.1016/j.ijmachtools.2013.06.007.
[11] Rauch M, Hascoet JY, Hamann JC, Plennel Y. A new approach for toolpath programming in incremental sheet forming. Int J Mater Form 2008;1:1191–4. https://doi.org/10.1007/s12289-008-0154-5.
[12] Henrard C, Bouffioux C, Eyckens P, Sol H, Duflou JR, Van Houtte P, et al. Forming forces in single point incremental forming: prediction by finite element simulations, validation and sensitivity. Comput Mech 2011;47:573–90. https://doi. org/10.1007/s00466-010-0563-4.
[13] Aerens R, Eyckens P, Van Bael A, Duflou JR. Force prediction for single point incremental forming deduced from experimental and FEM observations. Int J Adv Manuf Technol 2010;46:969–82. https://doi.org/10.1007/s00170-009-2160-2.
[14] Oraon M, Mandal S, Sharma V. Predicting the deformation force in the incremental sheet forming of AA3003. Materials Today: Proceedings 2021;45:5069–73. https://doi.org/10.1016/j.matpr.2021.01.578.
[15] Alsamhan A, Ragab AE, Dabwan A, Nasr MM, Hidri L. Prediction of formation force during single-point incremental sheet metal forming using artificial intelligence techniques. PloS One 2019;14:e0221341. https://doi.org/10.1371/journal. pone.0221341.
[16] Liu Z, Li Y. Small data-driven modeling of forming force in single point incremental forming using neural networks. Engineering with Computers 2020;36:1589–97. https://doi.org/10.1007/s00366-019-00781-6.
[17] Choudhary K, DeCost B. Atomistic line graph neural network for improved materials property predictions. Npj Comput Mater 2021;7:1–8. https://doi.org/ 10.1038/s41524-021-00650-1.
[18] Gasse M, Chetelat D, Ferroni N, Charlin L, Lodi A. Exact combinatorial optimization with graph convolutional neural networks. In: Advances in neural information processing systems. vol. 32. Curran Associates, Inc.; 2019.
[19] Mozaffar M, Liao S, Lin H, Ehmann K, Cao J. Geometry-agnostic data-driven thermal modeling of additive manufacturing processes using graph neural networks. Addit Manuf 2021;48:102449. https://doi.org/10.1016/j. addma.2021.102449.
[20] Chattopadhyay R, Ye J, Panchanathan S, Fan W, Davidson I. Multisource Domain Adaptation and Its Application to Early Detection of Fatigue 2011;6:717–25. https://doi.org/10.1145/2020408.2020520.
[21] AMPL | Facilities n.d. https://ampl.mech.northwestern.edu/facilities/index.html (accessed November 11, 2023).
[22] Capabilities | Machina Labs n.d. https://machinalabs.ai/capabilities (accessed November 11, 2023).
[23] Zhou Q-Y, Park J, Koltun V. Open3D: A Modern Library for 3D Data Processing 2018. doi:10.48550/arXiv.1801.09847.
[24] Veli˘ckovi´c P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y. Graph Attention ` Networks 2018. https://doi.org/10.48550/arXiv.1710.10903.
[25] Fey M, Lenssen JE. Fast Graph Representation Learning with PyTorch Geometric 2019. https://doi.org/10.48550/arXiv.1903.02428.
[26] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization 2017. doi:10.48550/arXiv.1412.6980.