Zebin Li
Industrial and Systems Engineering,
University at Buffalo, SUNY,
Buffalo, NY 14260
e-mail: zebinli@buffalo.edu

Luis Javier Segura
Assistant Professor
Industrial Engineering,
University of Louisville,
Louisville, KY 40292
e-mail: ljsegu01@louisville.edu

Yifu Li
Assistant Professor
Industrial Engineering,
University of Oklahoma,
Norman, OK 73019
e-mail: liyifu@ou.edu

Chi Zhou
Associate Professor
Industrial and Systems Engineering,
University at Buffalo, SUNY,
Buffalo, NY 14260
e-mail: chizhou@buffalo.edu

Hongyue Sun[1]
Assistant Professor
Industrial and Systems Engineering,
University at Buffalo, SUNY,
Buffalo, NY 14260
e-mail: hongyues@buffalo.edu

Multiclass Reinforced Active Learning for Droplet Pinch-Off Behaviors Identification in Inkjet Printing

Inkjet printing (IJP) is one of the promising additive manufacturing techniques that yield many innovations in electronic and biomedical products. In IJP, the products are fabricated by depositing droplets on substrates, and the quality of the products is highly affected by the droplet pinch-off behaviors. Therefore, identifying pinch-off behaviors of droplets is critical. However, annotating the pinch-off behaviors is burdensome since a large amount of images of pinch-off behaviors can be collected. Active learning (AL) is a machine learning technique which extracts human knowledge by iteratively acquiring human annotation and updating the classification model for the pinch-off behaviors identification. Consequently, a good classification performance can be achieved with limited labels. However, during the query process, the most informative instances (i.e., images) are varying and most query strategies in AL cannot handle these dynamics since they are handcrafted. Thus, this paper proposes a multiclass reinforced active learning (MCRAL) framework in which a query strategy is trained by reinforcement learning (RL). We designed a unique intrinsic reward signal to improve the classification model performance. Moreover, how to extract the features from images for pinch-off behavior identification is not trivial. Thus, we used a graph convolutional network for droplet image feature extraction. The results show that MCRAL excels AL and can reduce human efforts in pinch-off behavior identification. We further demonstrated that, by linking the process parameters to the predicted droplet pinch-off behaviors, the droplet pinch-off behavior can be adjusted based on MCRAL.
[DOI: 10.1115/1.4057002]

Keywords: additive manufacturing, inspection and quality control

1 Introduction

Additive manufacturing (AM) techniques draw great attentions recently. Among various AM techniques, inkjet printing (IJP) is undergoing enormous research development owing to its merits such as cost-effectiveness, high-resolution, and suitability to print different materials. Therefore, IJP shows great application potential in fabricating various electronic and biomedical products such as sensors and biochips [1–4].

Specifically, IJP deposits a sequence of micro-scale liquid-phase materials (i.e., droplets) on the substrate. Then, the deposition is solidified and the final products are formed [5]. Depending on the way that droplets drop off, IJP can be categorized into continuous dropping and drop-on-demand (DOD), and the latter method can achieve higher printing resolution [6]. In this paper, we focus on a popular DOD method, piezoelectric IJP, as shown in Fig. 1(*a*). During the piezoelectric IJP process, the ink (i.e., liquid material) fills the chamber, and the droplets are "squeezed" out from the nozzle when an electrical signal is applied to the piezoelectric actuator. The piezoelectric IJP process is governed by many controllable factors, such as process parameters (e.g., voltage and back-pressure) and ink properties (e.g., viscosity and density); and uncontrollable factors, such as environmental conditions (e.g., temperature, humidity) [7]. These factors together constitute a highly dynamic system and result in the difficulty of achieving consistent and stable droplets. The droplet behaviors can be categorized into droplet formation (e.g., droplet initiation, thinning, necking), droplet pinch-off, and droplet evolution. Among them, the droplet pinch-off behaviors play an important role in the droplet generation, which heavily affect the quality of the final printed products [8]. Therefore, identifying the droplet pinch-off behaviors is critical for guaranteeing the reliability and stability of the IJP process.

Based on the location of the first pinch-off position, four types of pinch-off behaviors can be specified, which are front pinching, hybrid pinching, exit pinching, and middle pinching, as shown in Fig. 1(*b*) [8]. Since the droplet pinch-off behaviors are critical for different applications, such as IJP and microfluidics, several empirical studies of the pinch-off behaviors have been conducted based on different material properties and application scenarios. For instance, Thiévenaz et al. studied the onset of heterogeneity in the pinch-off behaviors of suspension droplets [9]. Zhu et al. studied the dynamics of pinch-off behaviors under the influence of pressure fluctuations [10]. Roché et al. studied the effect of surface tension variations on the pinch-off behaviors with the presence of surfactants [11].

However, the actual droplet pinch-off behaviors can deviate from those expected under a combination of material properties and process parameters. Meanwhile, one can easily collect a large amount of images of droplets and try to train a machine learning (ML) model for classifying different droplet pinch-off behaviors. However, the training of the classification model requires labels of images, and annotating the images can be time-consuming for human experts. In addition, extracting features of pinch-off behaviors from images can be difficult since there are no well-defined features. Thus, the objective of this paper is to help researchers identify (i.e., classify) the droplet pinch-off behaviors by proposing a limited-label-required ML approach integrated with an automatic feature extraction method.

[1]Corresponding author.
Manuscript received November 7, 2022; final manuscript received February 15, 2023; published online March 15, 2023. Assoc. Editor: Cheryl Xu.

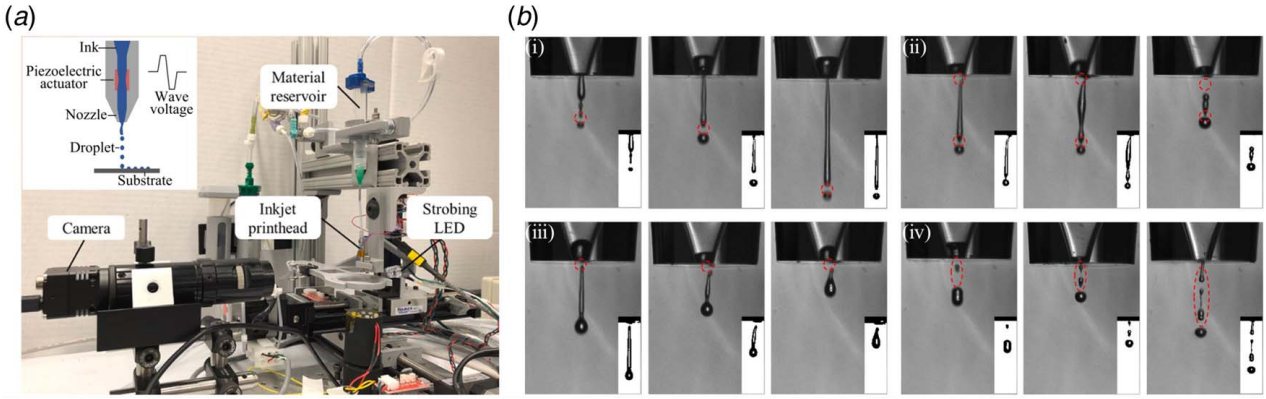

Fig. 1 The demonstrations of (*a*) piezoelectric IJP system (MicroFab Inc., reprinted from Ref. [5] with permission from Elsevier) and (*b*) different droplet pinch-off behaviors. The upper left figure in (*a*) shows the piezoelectric IJP process. The different droplet pinch-off behaviors in (*b*) are (i) front pinching, (ii) hybrid pinching, (iii) exit pinching, and (iv) middle pinching. The dashed circles are the locations where the pinch-off behaviors happen. The small binary images in the bottom-right are the processed images for the GCN feature extraction.

In this work, we adopt a graph convolutional network (GCN) to overcome the challenge of no well-defined features for droplet pinch-off behaviors. Graphs represent a data structure that can model objects in nodes and their relationships in edges. As a unique non-Euclidean data structure, a graph can model free-form data structures without extensive feature engineering [12], hence making the analysis highly adaptive across different morphologies in IJP. Specifically, graph-based convolution extracts multiscale localized spatial features over non-Euclidean domains and composes them to re-construct highly expressive representations.

Furthermore, given the limitation on data labels, we leverage active learning (AL) to iteratively acquire labels of droplet pinch-off behaviors from human experts and update the classification model. This process will terminate until the classifier reaches a satisfactory performance. Meanwhile, only a limited amount of instances (i.e., images) are expected to be annotated to yield a certain classification performance, which reduces the manual annotation effort. In AL, the query strategy plays an important role since it decides which instance to annotate. The commonly used handcrafted query strategies can be classified into heterogeneity-based, performance-based, representativeness-based, and hybrid approaches [13]. However, most of them are rigid and not suitable to the potential dynamics where the selected annotated instance may deviate from the most informative one during the annotating and updating iterations [14]. To compensate for the inflexibility of handcrafted query strategies, machine-learned query strategies draw researchers' attention recently. By considering the query process of AL as a sequential decision-making problem and modeling it as a Markov decision process (MDP), reinforcement learning (RL) is used to learn a query strategy directly from the data. Several works in different fields have utilized AL with RL-based query strategies, for instance, Refs. [14,15].

In this paper, we propose a multiclass reinforced active learning (MCRAL) framework for identifying (i.e., classifying) the droplet pinch-off behaviors in the IJP process. In particular, a multilayer perceptron (MLP) classifier is trained in a pool-based AL fashion. RL is used to train an intelligent agent to learn a policy based on the interaction with the environment, i.e., iteratively annotating unlabeled images and retraining the classifier. The deep Q-network (DQN) with several improvements including dueling structure, double Q-learning, and prioritized experience replay is used for training the agent [16–19]. An intrinsic reward is further proposed to improve the model capacity by forcing the agent to achieve some inherent goals (i.e., intrinsic motivation). The pool containing all of the unlabeled images is considered as the environment, and the policy determines the agent's actions in the environment. During the training process, each time after a (batch of) new unlabeled image(s) is annotated by human experts, the MLP classifier is retrained using all of the labeled images.

The designs of state, action, and reward are critical to RL. The state is used to characterize the unlabeled data pool (i.e., environment) and store the information of labeled images. In AL, the exploration and exploitation of the unlabeled data pool are considered as a dilemma and need to be jointly considered [20]. In this paper, graph density and margin are used to account for them, respectively. The former captures the similarity of images in the unlabeled data pool by a graph structure, and the latter captures the uncertainty of the images in the unlabeled data pool. Besides, the labeled images are also considered in the state to restrict their repeated selections. The action of the agent is to update the current labeled images by selecting an unlabeled image and annotating it with the help of human experts in each query iteration. The reward measures the goodness of actions and we use the difference of the cross-entropy between two consecutive actions as a reward. Since the agent will receive the reward from the environment during each query iteration, we refer it as an extrinsic reward. Besides, we further propose an intrinsic reward that is only assigned to the agent when some conditions are satisfied (see details in Sec. 4.5). This intrinsic reward mimics the "self-rewarding" phenomenon. The case study shows the effectiveness of the intrinsic reward in improving the classifier performance. The final reward consists of the extrinsic and intrinsic rewards. By maximizing the expected cumulative reward during the training, the agent can learn to take actions (i.e., update the current labeled images) that improve the classifier performance ultimately.

To illustrate the advantage of the proposed method (i.e., MCRAL with intrinsic reward), we compare it with several benchmarks including unsupervised ML (i.e., *k*-means), the classical AL with uncertainty sampling, MCRAL without intrinsic reward, and supervised ML (i.e., the same classifier trained by all the data from the training set). The results suggest that compared to AL with uncertainty sampling, MCRAL achieves a better performance and the intrinsic reward can further enhance the model performance. Unsupervised ML achieves the worst classification result and supervised ML achieves the best classification result as expected, since no label and all labels are required, respectively. Finally, by linking the process parameters to the droplet pinch-off behaviors predicted from the proposal MCRAL classifier, we demonstrate the droplet pinch-off behavior adjustments based on MCRAL.

The main contributions of this paper include:

(1) We extract features of IJP images by GCN which considers the multiscale localized spatial features among pixels of images.

(2) We propose an AL framework with RL-based query strategy to compensate for the rigidity of handcrafted query strategies.
(3) We introduce the novel intrinsic reward in reinforced active learning (RAL) framework for multiclass image classification.
(4) We apply the proposed method to the droplet pinch-off behavior adjustments based on the learned MCRAL.

The reminders of the paper are organized as follows. We first review related works in Sec. 2. Then, in Sec. 3, we introduce the proposed method. In Sec. 4, the experimental details and results are presented. Finally, we conclude the paper in Sec. 5.

2 Literature Review

2.1 Inkjet Printing Droplet Behaviors. The IJP process performance is substantially dependent on the droplet behaviors [5]. Different behaviors will produce different formations of the printed parts; hence, affecting the product consistency and repeatability [21]. However, identifying these behaviors, for example, droplet formation and droplet pinch-off behaviors is difficult to achieve. Several empirical, analytical, and ML methods have been explored to study the IJP droplet behaviors for process performance enhancement [22–25].

Considerable attention has been given to empirical numbers, namely Onhesorge (Oh), Weber (We), and Reynolds (Re) to analyze the IJP process jettability. For instance, Jiao et al. studied the influence of ink properties and voltage parameters on piezoelectric inkjet droplet formation [22]. Here, the droplet behaviors are correlated to material rheological properties and dimensionless empirical numbers (i.e., Oh, We, and Re). Xu et al. developed a phase diagram of pinch-off behaviors for piezoelectric IJP [8]. Four types of pinch-off behaviors were devised for different alginate solution concentrations. Although beneficial to understand the droplet behaviors in IJP, the empirical methods are not suitable for large-scale and fast droplet behaviors classification.

Analytical methods have also been implemented to study the IJP droplet behaviors. The majority of the studies have focused on solving the Navier–Stokes equations via the volume of fluid approach [23,26]. For instance, Tofan et al. modeled the droplet motion and interaction with flat surfaces in the IJP process [23]. Besides, special attention has also been given to the lattice Boltzmann method. Zhang et al. proposed a numerical investigation of multidroplets deposited lines' morphology with a multiple-relaxation-time lattice Boltzmann model [27]. These methodologies have shown potential to analytically describe the droplet behaviors in IJP, but they are computationally inefficient and do not consider the uncertainties that are present in the real-world process.

Lately, ML methodologies have been demonstrated in the IJP process. Andalib et al. used classification and regression methods to analyze the time-dependent behavior of a methanol droplet at different levels of environmental humidity and temperature of the substrate [28]. Wang et al. tried to estimate the droplet volume in IJP by a data-driven autoregressive exogenous (ARX) model with the help of a droplet volume adjuster [29]. They modeled the droplet pinch-off instants and droplet velocities in polynomials with respect to the drive waveform. Huang et al. proposed a spatiotemporal fusion network to identify the droplet formation behaviors in the IJP process using video data [24]. Two streams of the networks were used to deal with spatial and temporal features of droplet formation behaviors, respectively.

It can be seen that though there are some empirical studies of the droplet pinch-off behaviors in the IJP process, the identification of them through ML models with limited labels is not studied.

2.2 Graph Convolutional Networks for Image Classification. Extracting proper features plays an important role in image classification tasks, and researchers usually use user-defined features to incorporate domain knowledge [30] or use convolutional neural networks (CNNs) for general feature extraction [31]. However, the former one is limited to specific tasks, and the latter one suffers from the fixed number of neighboring pixels for one pixel and the fixed scanning order when conducting convolution operation, which cannot deal with spatial relationships among different pixels [32].

The power of tackling irregular spatial information enables graph convolutional networks (GCNs) to address the aforementioned difficulty that CNNs faced. Extensive studies have been performed on GCN-based hyperspectral image classification. For instance, Hong et al. proposed a minibatch GCN to address the large-scale training issue of GCNs [33]. They used GCN to extract the middle- and long-range spatial correlations between samples. Since CNNs can extract different features (i.e., short-range spatial correlations) compared with GCNs, they also tried to fuse CNN and GCN features to further improve the model capacity. Mou et al. proposed a nonlocal GCN framework, of which the whole hyperspectral image was represented as a nonlocal graph [34]. Specifically, each vertex in the graph represented a pixel in the image. Then, the graph convolution was performed on the nonlocal graph. In this paper, we use GCN to exact features for the morphologies of the droplet pinch-off behaviors.

2.3 Reinforced Active Learning. AL is getting more and more attentions due to its ability to reduce annotating efforts in different domains, such as image classification [35], experimental design [36], and manufacturing field [37,38]. For instance, Yue et al. incorporated the measurements' uncertainties into AL to improve the model prediction capacity for the shape control of composite fuselage [37]. Lee et al. developed a new AL method considering the physics constraints of engineering systems [38]. In the classical AL framework, the query strategy is always fixed and may not be appropriate during the dynamic query process. Many researchers try to address this issue by combining RL with AL. This is because AL can be viewed as a sequential decision-making task whereas RL can be used as a query strategy. Several works have been proposed to demonstrate the feasibility of combining RL with AL. For instance, Liu et al. studied the person re-identification task by performing RAL [39]. They used a graph structure captured the similarity of images as the state, and selecting the next image was considered as the action. The reward was a hard triplet loss which measured the uncertainty value of the selected image. Casanova et al. used RAL for image segmentation [40]. The predictions and the prediction uncertainties of the segmentation model on a representative subset were used as the state. The action was to annotate a pixel-wise unlabeled region. The improvement of the segmentation performance measured by Intersection-over-Union on another subset after updating the segmentation model was used as the reward signal.

The combination of RL with AL also has been applied to other tasks, for example, anomaly detection [41]. Since different tasks require different designs of state, action, and reward, there is no universal RAL framework. Besides, inspired by how human beings or animals learning skills, the concept of intrinsic motivation is introduced in RL to encourage the agent exploration [42]. Some studies have incorporated an additional intrinsic reward to help the agent obtain more information of the environment, such as Refs. [43,44]. However, the intrinsic motivation has not been introduced in RAL works reviewed above.

In summary, the identification of droplet pinch-off behaviors in the IJP process has not been well studied yet. To address the drawbacks that the existing supervised ML methods rely on a large number of labels and there are no well-defined features to describe the droplet pinch-off behaviors from images, we perform GCN to extract features and propose an MCRAL framework enhanced by the intrinsic reward for the pinch-off behaviors classification.

3 Multiclass Reinforced Active Learning

3.1 Overview of the Proposed Framework. Figure 2 shows the proposed MCRAL framework. After data collection, we first

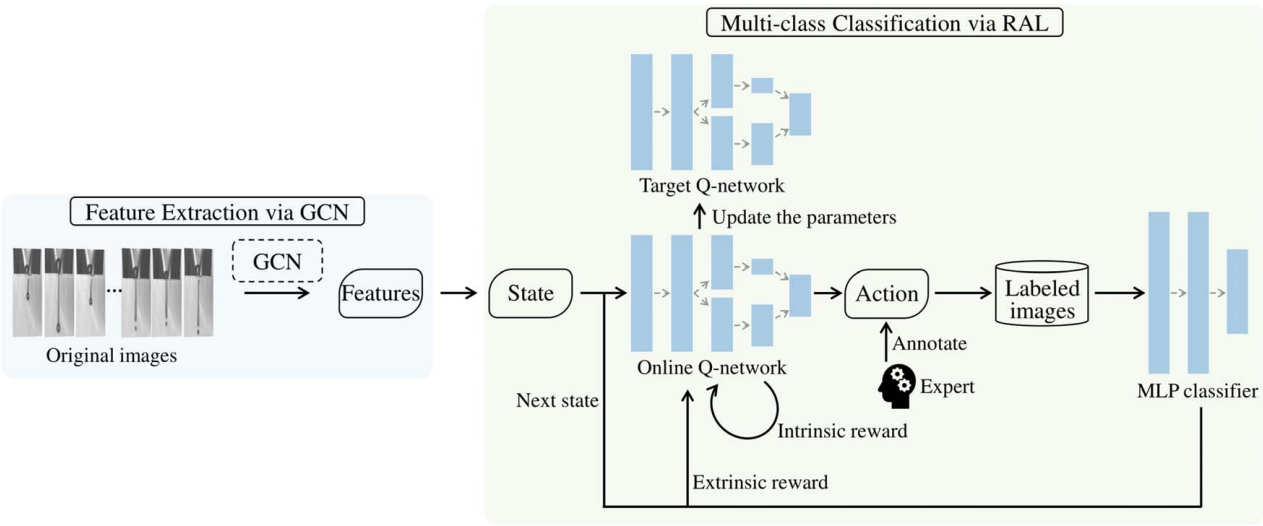

Fig. 2 The proposed MCRAL framework

perform feature extraction on the collected images by using GCN (see details in Sec. 4.1). Then, we perform MCRAL to train a multi-class classifier for the pinch-off behaviors identification task. The agent will select the most informative image during each iteration. Since DQN has shown its success for the tasks with discrete action space, we develop a variant of DQN which incorporates several improvements including dueling structure to improve the estimation precision, double Q-learning to mitigate the overestimation, and prioritized experience replay to train the agent [16–19].

Algorithm 1 Multi-class Reinforced Active Learning (MCRAL)

Input : The environment X_u and the query budget
Output: Updated MLP classifier
Initialize: The MLP classifier, the state, and the reward

for *each query process* **do**
  for *each query iteration in the query process* **do**
    Select x_i from X_u following ε-greedy policy and receive its label from human expert;
    Update the MLP classifier using all the current labeled images;
    Observe S_{t+1} according to Eqs. (7) - (8) and R_t according to Eqs. (9) - (13);
    Save the tuple $\langle S_t, A_t, R_t, S_{t+1}\rangle$ into the replay buffer;
  for *each update step of Dueling Double DQN* **do**
    Sample a minibatch of tuples from the replay buffer according to priority;
    Calculate the target according to Eq. (6);
    Perform stochastic gradient descent to solve Eq. (3);
    Update the parameters of the online Q-network;
    Update the parameters of the target Q-network every τ steps.

As shown in Algorithm 1, the proposed framework begins with several initialization including initializing the classifier (i.e., MLP), the state, etc., with the randomly selected unlabeled images. Next, during each query iteration in a query process (i.e., exhaust the query budget), the agent will take action A_t, i.e., select an unlabeled image following its policy π and annotate it as either "front pinching," "hybrid pinching," "exit pinching," or "middle pinching." After the classifier is updated by using all the current labeled images, the agent will observe a new state S_{t+1} and a reward signal

R_t. Then, the transitions, i.e., S_t, A_t, R_t, S_{t+1}, will be saved into the replay buffer. Once exhausting the query budget, the query process will be terminated. Meanwhile, the agent will update its parameters by performing stochastic gradient descent. The algorithm ends until the predefined number of query processes is reached. The details of the proposed framework will be introduced as follows.

3.2 Active Learning. AL is a type of ML technique that aims at using a small number of labels to train a model with good performance [13]. It provides a general framework to involve human intervention during model training for both classification and regression tasks. Specifically, AL is comprised three components, model (i.e., classifier or regressor), query strategy, and human expert. Model can be any supervised ML method and it is selected or developed based on the certain task. Query strategy is the core of AL and it is responsible for selecting the most informative instances to maximize the model's improvement. Human expert helps with annotating the instances selected from the query strategy. Because of the ability of AL that reduces the required amount of labeled data for training a model, it gains great attention in many fields in which annotating costs are high. We utilize the AL framework for identifying the droplet pinch-off behaviors and integrate RL into it as the query strategy, which will be introduced later.

3.3 Graph Convolutional Network. Graphs are a kind of data structure for dealing with unstructured data via representing data points by a set of nodes and their relationships by edges. GCN first transforms the graph to the spectral domain by conducting the graph Fourier transform and performing the convolution operation. Then, the results are transformed back from the spectral domain by conducting the inverse graph Fourier transform [45].

Given an undirected graph G with N nodes, where each node has a D-dimensional feature vector, the node feature matrix of G can be formed as $X \in \mathbb{R}^{N\times D}$. Besides, an adjacency matrix A of the graph G characterizes the relationships among nodes. Then, considering a multilayer GCN model, its layer-wise propagation rule is defined as [45]

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \tag{1}$$

where $\tilde{A} = A + I_N$. I_N is the identity matrix with the dimension of $N\times N$. $\tilde{D}$ is the degree matrix of $\tilde{A}$ and is calculated as $\tilde{D}_{jj} = \sum_j \tilde{A}_{ij}$. $H^{(l)} \in \mathbb{R}^{N\times D}$ is the extracted features of G in the lth layer of GCN and $H^{(0)} =$

X. $W^{(l)}$ is the weight matrix for the lth layer. $\sigma(\cdot)$ is the activation function, for example, rectified linear unit (ReLU).

Thanks to the capacity of GCN in extracting irregular spatial information, we apply GCN to each droplet pinch-off behavior image to obtain the droplet pinch-off morphology features. These morphology features will be used in the later MCRAL framework.

3.4 Deep Reinforcement Learning. The success of DQN that achieved human-level control in several video games draws great attention [16]. By combining neural networks with a classical RL algorithm, i.e., Q-learning, DQN is able to handle high dimensional action/state space. Specifically, DQN is comprised two Q-networks with the same network structure, which are named as online Q-network and target Q-network, as shown in Fig. 2. The online Q-network $Q(S_t, A_t; \boldsymbol{\theta}_t)$ is used to calculate Q-values at each time-step t. The target Q-network is used to freeze the target value when training the online Q-network via stochastic gradient descent. The target is defined as [17]

$$Y_t^{DQN} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t^-) \tag{2}$$

where γ discounts the future rewards. $\boldsymbol{\theta}_t^-$ is the parameter copied from $\boldsymbol{\theta}_t$ in online Q-network every τ steps, and a is the selected action. The loss function for training the online Q-network is defined as

$$L(\boldsymbol{\theta}_t) = (Y_t - Q(S_t, a; \boldsymbol{\theta}_t))^2 \tag{3}$$

where $Y_t = Y_t^{DQN}$. The calculated Q-values are used to measure how good (or bad) to take an action under the current state. Then, the policy $\pi(A_t|S_t)$ is derived from the Q-values. In particular, DQN uses ε-greedy policy which randomly selects an unlabeled image with probability ϵ or selects an unlabeled image according to the maximum Q-value with probability $1-\varepsilon$. Besides, an experience replay is performed for DQN which saves the collected transitions (i.e., S_t, A_t, R_t, S_{t+1}) during the online Q-network training. By randomly sampling a batch of transitions from the experience replay to train the online Q-network, the correlations in the transitions can be reduced. The experience replay also enables the agent to learn from the preceding policies.

Though DQN has a powerful capacity, several improvements are proposed to further enhance it. Double DQN has the same network structure as DQN but only changes the target definition as [17]

$$Y_t^{DDQN} = R_{t+1} + \gamma Q\left(S_{t+1}, \arg\max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}_t^-\right) \tag{4}$$

The loss function for training the agent is the same as DQN (i.e., Eq. (3)), other than replacing Y_t with Y_t^{DDQN}. It incorporates the idea from double Q-learning which uses two different networks to evaluate and select actions. This decoupling of the evaluation and the selection of actions mitigates the overestimation problem of DQN, where the max operator uses the same network to select and evaluate an action.

Dueling structure further improves the estimation of Q-values via replacing the last fully connected layer of the Q-networks by two fully connected layers which are used to separately estimate the state value function and the advantage function [18]. The intuition of the dueling structure is that the estimation of each action may not be necessary at some time-steps. By aggregating the two estimates of the state and the advantage, the estimation of the Q-values can be more precise. Specifically, it is defined as [18]

$$Q(S_t, a; \boldsymbol{\theta}_t, \boldsymbol{\alpha}_t, \boldsymbol{\beta}_t) = V(S_t; \boldsymbol{\theta}_t, \boldsymbol{\beta}_t) + A(S_t, a; \boldsymbol{\theta}_t, \boldsymbol{\alpha}_t) - \frac{1}{|A|}\sum_{a'} A(S_t, a'; \boldsymbol{\theta}_t, \boldsymbol{\alpha}_t) \tag{5}$$

where $V(S_t; \boldsymbol{\theta}_t, \boldsymbol{\beta}_t)$ and $A(S_t, a; \boldsymbol{\theta}_t, \boldsymbol{\alpha}_t)$ represent the state value function and the advantage function, respectively. $\boldsymbol{\alpha}_t$ and $\boldsymbol{\beta}_t$ are the parameters of the two fully connected layers.

The prioritized experience replay improves the way that DQN samples transition [19]. In DQN, the transitions are randomly sampled which ignores their priority. However, the transitions are not of the same importance for training the agent. The main idea of the prioritized experience replay is to increase the sampling probability of the transitions from whom the agent can learn more. This results in a faster learning speed and a better policy.

By combining all the above-mentioned improvements with the original DQN, we get our final target Q-value defined as

$$Y_t^{\text{target}} = R_{t+1} + \gamma Q\left(S_{t+1}, \arg\max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t, \boldsymbol{\alpha}_t, \boldsymbol{\beta}_t); \boldsymbol{\theta}_t^-, \boldsymbol{\alpha}_t^-, \boldsymbol{\beta}_t^-\right) \tag{6}$$

where $\boldsymbol{\alpha}_t^-$ and $\boldsymbol{\beta}_t^-$ are the parameters of dueling structure when combining with double Q-learning.

3.5 State, Action, Reward. We introduce the details of the designs of state, action, and reward in the following.

3.5.1 State. The state consists of two parts, the characterization of the unlabeled data pool by graph density and margin, and an indicator vector indicating whether the image is annotated or not. The former one is used to select the next unlabeled image to be annotated, and the latter one is used to restrict the repeated selections of the already labeled images. We introduce the details next.

Graph density (*Gra*) captures the exploration of the unlabeled data pool by introducing a k-nearest neighbor graph, where k represents the amount of nearest neighbors and each node of the graph represents one image. *Gra* is defined as [20]

$$Gra(x_i) = \frac{\sum_i W_{ij}}{\sum_i H_{ij}} \tag{7}$$

$x_i \in \mathbb{R}^d$ denotes the features of image i in the unlabeled data pool. $H_{ij} = \max(\hat{H}_{ij}, \hat{H}_{ji})$ where $\hat{H}_{ij} = 1$ indicates the Manhattan distance $d(x_i, x_j)$ between x_i and x_j is among the k smallest distances of x_i. W_{ij} is calculated as $W_{ij} = H_{ij}\exp(-d(x_i, x_j)/2\sigma^2)$. The graph density is updated by $Gra(x_j) = Gra(x_j) - Gra(x_i)H_{ij}$ [20]. This operation reduces the weights of the directly connected neighbors for the selected x_i. Thus, it avoids the repeated selection from the same region of the unlabeled data pool.

Margin (*Mar*) characterizes the exploitation of the unlabeled data pool based on the prediction uncertainty since instances with larger uncertainty are near the classification boundary. Considering the predicted labels of image i for multiclass classification task with the highest probability $\hat{y}_{i1}$ and the second-highest probability $\hat{y}_{i2}$, margin is defined as [20]

$$Mar(x_i) = P(\hat{y}_{i1}|x_i) - P(\hat{y}_{i2}|x_i) \tag{8}$$

Besides, an indicator vector of the same length of the unlabeled data pool is used to keep track of which of them are being labeled, and to avoid the repeated selections of them. If one image is being labeled, then the corresponding entry of the indicator vector equals to 1, otherwise, 0.

Finally, the graph density, the margin, and the indicator vector together constitute the state, which is a three-column matrix and the number of rows is the same as the number of images.

3.5.2 Action. We refer actions to the updates of current labeled images by first selecting an unlabeled image from the unlabeled data pool according to the query strategy, and then annotating it by human experts. It is worth noting that the initial action space is the same as the unlabeled data pool. Since the unlabeled images can only be selected and annotated once, the indicator vector in the current query process is used to restrict the repeated selections. Once the query number reaches the budget, the query process will be terminated.

3.5.3 Reward. RL learns a good policy by maximizing the cumulative discounted future reward, thus, the design of the reward is of importance to achieve a good model performance. Here, the reward is defined as

$$R_t = R_t^{\text{extrinsic}} + R_t^{\text{intrinsic}} \tag{9}$$

where $R_t^{\text{extrinsic}}$ represents the extrinsic reward the agent received from the environment at each time-step t (i.e., during each query iteration), and $R_t^{\text{intrinsic}}$ denotes the proposed intrinsic reward, where the intuition is the "self-rewarding" of the agent.

The extrinsic reward is calculated as the difference of the cross-entropy on a validation data set between two consecutive query iterations:

$$R_t^{\text{extrinsic}} = CE_{t-1} - CE_t \tag{10}$$

where $CE = -\sum_{c=1}^{C} r_{i,c} \log(q_{i,c})$ is cross-entropy. Here, C represents the number of different classes and $r_{i,c}$ is a binary indicator which equals to 1 if the classification for the instance i of class c is correct, otherwise, 0. $q_{i,c}$ is the prediction probability for the instance i of class c. The utilization of the validation data set for calculating the extrinsic reward can also be seen in other RAL works, for example, Refs. [15,40]. Since the extrinsic reward will be positive if the cross-entropy is reduced, therefore, it will promote the agent to take actions (i.e., select unlabeled images and annotate it) that can improve the classification performance.

Besides, the agent will receive an intrinsic reward if some conditions are satisfied during each query iteration. To define this intrinsic reward, let us first visit some terms to quantify the classification performance, i.e., $F1$, $F1^{\text{weighted}}$, and $F1^{\text{min}}$. $F1$ represents $F1$ score and is calculated as

$$F1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{11}$$

where precision = true positive/true positive + false positive and recall = true positive/true positive + false negative, respectively. $F1^{\text{weighted}}$ represents weighted F1 score and it considers sample imbalance:

$$F1^{\text{weighted}} = \sum_{i}^{C} r_i * F1_i \tag{12}$$

where r_i is the ratio of instances for each class among the total instances, and $F1_i$ is the $F1$ score for the ith class. $F1^{\text{min}}$ is the minimum $F1$ score among all classes.

Since the pinch-off behaviors identification task is a multiclass classification task, we are not only interested in the overall performance of the classifier but also its lower-bound performance. Thus, we use $F1^{\text{weighted}}$ to measure the overall classifier performance and $F1^{\text{min}}$ to measure the worst performance of the classifier on one class. Then, the intrinsic reward is defined as

$$R_t^{\text{intrinsic}} = \begin{cases} m, & F1^{\text{weighted}}(t) \geq n,\ F1^{\text{min}}(t) \geq z \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Here, m is the amount of the intrinsic reward. $F1^{\text{weighted}}(t)$ represents the $F1$ score weighted by the number of instances in each class of the validation data set during query iteration t. $F1^{\text{min}}(t)$ is the minimum $F1$ score among all of the classes of the validation data set during query iteration t. n and z refer to the thresholds of $F1^{\text{weighted}}(t)$ and $F1^{\text{min}}(t)$, respectively. See details about the mechanism of the proposed intrinsic reward as well as the selections of m, n, and z in Sec. 4.5.

3.6 Multilayer Perceptron Classifier. MLP is a kind of artificial neural networks which mimics the biological neural networks [46]. It consists of three different layers of neurons, including input layer, hidden layer, and output layer. During the calculation, the nonlinear activation function enables the MLP to capture the nonlinearity of the input data, which improves the inference capacity of the MLP. MLP is fully connected, in other words, all the neurons in the current layer are connected to all the neurons in the following layer with certain weights. MLP is trained in a back-propagation way, in which the gradient of the loss function is used to update the weights (and the biases) of the neurons. MLP is flexible and powerful since the number and the breadth of the hidden layers can be modified to adapt to different data complexity.

Table 1 The ranges of the IJP process parameters for data collection

Process parameters	Low levels	High levels
Back-pressure (in H_2O)	−6	−3
Dwell voltage (V)	48.1	70
Dwell time (μs)	28.2	35.8
Echo voltage (V)	−70	−48.1
Echo time (μs)	56.5	71.6

4 Case Study

4.1 Data Acquisition and Preprocessing. We collect the images by performing the experiments under diverse combinations of process parameters shown in Table 1. As shown in Fig. 1(*b*), the droplets are "squeezed" out from the micro-dispensing nozzle (100 μm diameter) when an electric signal with trapezoidal waveform is applied to the piezoelectric actuator. In total, 94 experiments with different combinations of parameters are conducted. A CCD camera (Sensor Technologies Inc.) enhanced by a magnification lens is used to capture the droplet images with a resolution of 480×640 pixels. For one experiment, the image stream includes droplet generation, pinch-off, and evolution. Since we focus on the pinch-off phenomenon, only the frames where the pinch-off phenomenon happens are collected. In total, we obtain 436 images including 157 exit pinching, 150 front pinching, 81 hybrid pinching, and 48 middle pinching. The original data set can be accessed at the link shown below.[2]

After data collection, the acquired images are processed and go through feature extraction. First, the original images are resized and binarized to reduce the number of pixels and remove the background (see Fig. 1(*b*)). Then, a graph based on the pixel coordinate pairwise Manhattan distance is built for each processed image. Intuitively, the pixels that have a smaller distance on the image should have a stronger connection on the graph. Therefore, for each image, the adjacency matrix (A) of the corresponding graph is calculated as $A(i, j) = \exp((|i_1 - j_1| + |i_2 - j_2|)^{1/2})$, where i represents the ith pixel of the image, and i_1 and i_2 represent the x and y coordinates of the pixel. Also, we define $A(i, j) = 0$, if $A(i, j) < 0.0001$ or $A(i, j) = 1$. By doing so, the pixels that are too far away from each other will be considered as having no linkage, and the pixels have no linkage with themselves.

It should be noticed that the intuition of converting original pinch-off images to graphs is that different droplet pinch-off behaviors have no explicit features but have similar topologies (i.e., geometric features). Since GCN is powerful to deal with nonstructural data (e.g., graph), therefore, we next perform GCN on the built graphs to extract geometric features. Specifically, a GCN with two convolutional layers and one global sum pooling layer is conducted on each graph to extract features from each processed image. 256 channels and the ReLU activation function are used in each convolutional layer. By stacking multiple convolutional layers, the center nodes can be updated by the distant nodes with no direct linkage. As a result, GCN is able to extract high-level abstract features of the graph.

[2] https://stream.eng.buffalo.edu/public_data/Droplet_Dataset.zip

To fully utilize the label information of the validation data set, it is not only used for calculating the rewards but also used for training the GCN. Specifically, the validation data set which includes five images for each class is randomly split into GCN training and GCN testing data set by the ratio of 6:4 every 50 GCN training iterations. Such a re-splitting of the validation data set every 50 iterations will exploit full information of the validation data set to train the GCN. Then, an MLP classifier with two hidden layers (128 neurons in each layer and ReLU as the activation function) is trained on the GCN training data set every 10 GCN training iterations. During other GCN training iterations, the weights of the MLP classifier are fixed. This delayed update of the MLP classifier avoids the oscillation of the loss for training the GCN. The cross-entropy loss of the MLP classifier on the GCN testing data set is used to update the GCN weights via stochastic gradient descent. Both the learning rates for training the GCN and the MLP classifier are set to 0.001.

After 1000 GCN training iterations, the GCN features that result in the highest weighted F1 score on the GCN testing data set are used for the following MCRAL experiments. Finally, we get 256 features for each graph (i.e., image).

It should be pointed out that though some works visualized GCN features to help interpret their results, in most of them, the nodes have features that support node-level interpretation and enable feature visualization [47–49]. However, in our case, the nodes in the graphs are the pixels in the corresponding images, thus, those nodes do not have node features. Besides, the obtained GCN features are high-level abstract and each entry in the feature vector does not have physical meaning. Therefore, the visualization of these features is challenging.

4.2 Experimental Details. We implement our proposed MCRAL framework based on modAL [50] and Tianshou [51]. The main parameters of MCRAL are shown in Table 2. The Adam optimizer and ReLU activation function are used for the Q-networks in RL.

For conducting the experiment, we randomly split the images excluding the validation data set into the training data set and the testing data set by the ratio of 8:2. 8 randomly selected images (i.e., two images from each class) from the unlabeled data pool are used for the initialization of the MLP classifier (i.e., two hidden layers with 128 neurons in each layer and ReLU as the activation function). 30 query iterations are conducted for one query process and five query processes in total are conducted for the training of MCRAL. We repeat the above procedures for 50 replications and report the average and standard deviation of the classification results over replications.

To fairly compare MCRAL (i.e., MCRAL without intrinsic reward) with AL (i.e., AL with uncertainty sampling), we allow AL to obtain the same number of labels and be initialized by the same images as MCRAL does. Then, the classifier is trained based on all the labeled images including those in the validation data set, the initialization images, and the queried images. The classification performance is obtained by applying the classifier to the testing data set. To examine the contribution of the intrinsic reward, we directly incorporate it ($m = 0.5$, $n = 0.45$, and $z = 0.2$, see Sec. 4.5 for justifications) in MCRAL without intrinsic reward with other setup remains the same. The added intrinsic reward will affect the selection of unlabeled images as well as the number of required labels.

Table 2 Parameters of MCRAL

Parameters	Values
Number of hidden layers	3
Number of neurons in each hidden layer	256, 256, 256
Number of nearest neighbors in graph density (k)	10
Query budget	30
Learning rate for the agent	0.001
Discount factor	0.99
Batch size for updating the agent	8
Size of the prioritized replay	500
Prioritization exponent of the prioritized replay	0.5
Importance sample soft coefficient of the prioritized replay	0.5
ε for the ε-greedy policy	0.1
Agent update frequency (τ)	10

4.3 Experimental Results. As mentioned above, we first obtain tuned GCN features of images and then conduct the experiments. Specifically, we compare our proposed MCRAL framework (i.e., MCRAL with intrinsic reward, denoted as MCRAL_intrinsic) with several benchmarks including AL with uncertainty sampling (denoted as AL) and MCRAL without intrinsic reward (denoted as MCRAL). Furthermore, unsupervised ML (i.e., clustering by k-means) and supervised ML (i.e., exhausting all the labels of training data set) are applied to the data set, which serve as the lower bound and the upper bound of the classification performance, respectively. For unsupervised ML, since the labels and the clusters are not one-to-one correspondence, its result is selected as the highest weighted F1 score among all the label combinations [30]. Besides, to demonstrate the effectiveness of training GCN using the validation data set, we perform the same supervised ML as above using the untuned GCN features (i.e., the features extracted with randomly initialized GCN weights) and compare it with supervised ML using tuned GCN features.

The classification results and label utilization comparisons are shown in Fig. 3. The boxplots are the weighted F1 score for different methods over 50 replications. The triangles and the short horizontal lines represent the means and the medians of the weighted F1 score, respectively. The histogram shows the average number of labels used by each method. We can observe that unsupervised ML has the worst classification result since no label is used. MCRAL achieves a higher weighted F1 score than AL when the same number of labels are used due to its ability of handling the dynamics during the query process. The intrinsic reward can further improve the classifier performance. Supervised ML reaches the best classification performance as expected since it requires annotating all of the unlabeled images. On the contrary,

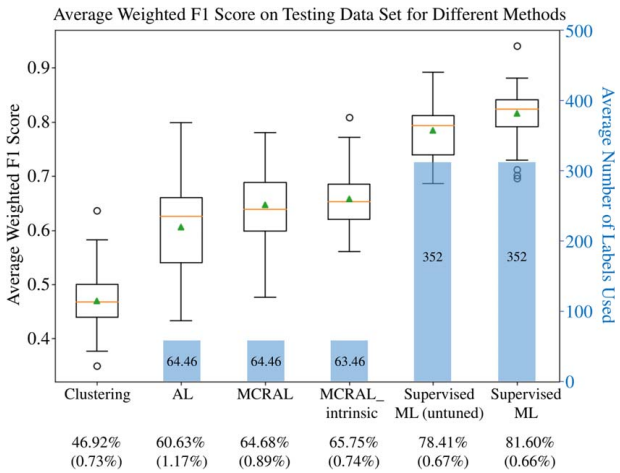

Fig. 3 Model performance and number of labels used for different methods. The boxplots are the weighted F1 score for each method, and for the clustering, it is the highest weighted F1 score among all the possible label combinations. The histogram is the number of required labels used in training each method. The last two columns show the results of the same supervised ML setting by using untuned GCN features and tuned GCN features, respectively. The exact weighted F1 score and the standard deviation (i.e., numbers in the parentheses) are shown in the figure.

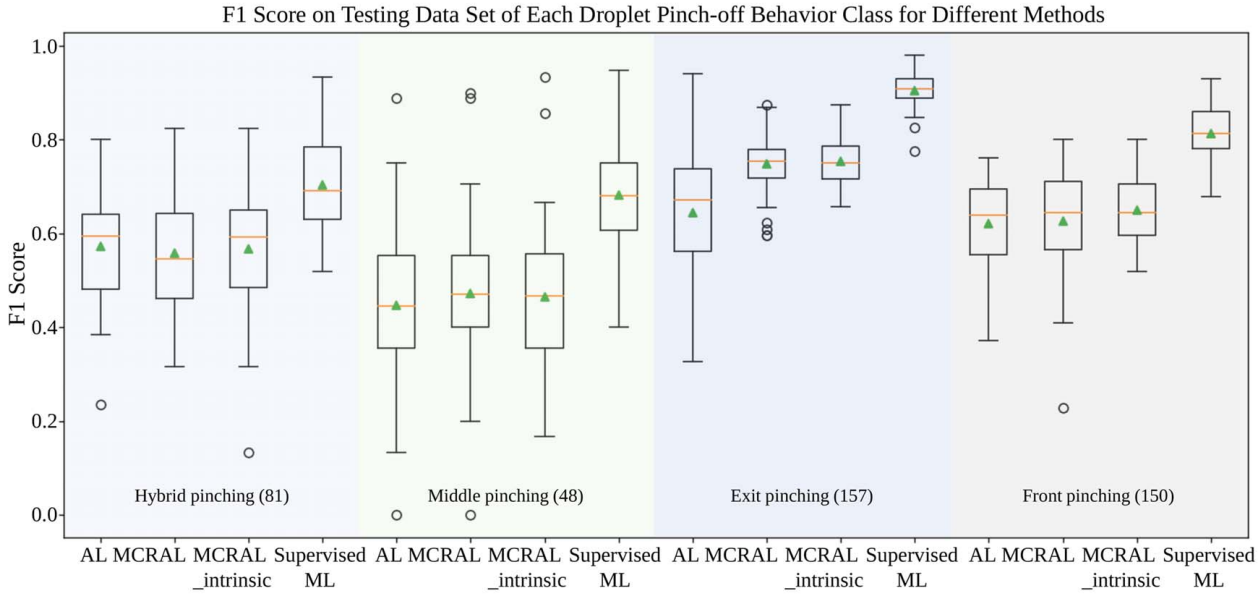

Fig. 4 The F1 score on the testing data set of each droplet pinch-off behavior class for different methods with 50 replications. Different colors represent different droplet pinch-off classes, and the numbers in the parentheses are the number of images of that class. In each class, the boxplots from left to right show the classification results of AL, MCRAL, MCRAL with intrinsic reward, and supervised ML, respectively.

only around 18% (including the validation data set) labels of the unlabeled images are required by MCRAL. The results show the ability of MCRAL to achieve a relatively good classification performance with a limited number of annotated labels. Besides, the fifth column is the result for supervised ML using untuned GCN features. By comparing it with the last column, it shows the effectiveness of training GCN by using the validation data set.

The detailed classification performance of AL, MCRAL, MCRAL with intrinsic reward, and supervised ML on the testing data set of each droplet pinch-off behavior class is shown in Fig. 4. In Fig. 4, each box represents one droplet pinch-off behavior class and the numbers in the parentheses are the number of images in that class. The boxplots represent the F1 score of different methods with 50 replications under the same experimental setting described above. It can be seen that all of these four methods have a higher F1 score for the exit pinching and the front pinching, compared to the hybrid pinching and the middle pinching. This is because the numbers of images of the exit pinching and the front pinching classes are much larger than the hybrid pinching and the middle pinching classes. In addition, the phenomenon that the exit pinching and the front pinching only have one pinch-off position while the middle pinching and the hybrid pinching have multiple pinch-off positions leads to the easier classification of the former two pinch-off classes and more classification difficulty in the later two pinch-off classes. Besides, compared to AL, the main improvement of MCRAL and MCRAL_ intrinsic also occurs in the exit pinching and front pinching, where the F1 score is increased dramatically. This phenomenon suggests that the RL-based query strategy is able to dynamically select the instances that benefit the classifier most. The supervised ML classification results show the upper bound F1 score for each class.

4.4 Pinch-Off Behavior Adjustment Based on Trained MCRAL Classifier. In this section, we further apply the trained MCRAL classifier to the inkjet printing droplet pinch-off behavior adjustments, by linking the IJP process parameters to the droplet pinch-off behaviors.

To achieve this, we first predict labels of the testing data set by using the trained MCRAL classifier. Then, we build another MLP classifier to link the process parameters (i.e., back-pressure, dwell voltage, and dwell time) to droplet pinch-off behaviors. In addition to the testing data set, all of the available labels during the query process are also used including validation data set and queried data set. This MLP classifier is learned by stochastic gradient descent. Then, one can adjust the process parameters to change the pinch-off behaviors according to the learned MLP classifier. As an illustration, Fig. 5(*a*) shows the phase diagram at back-pressure of −4.5 (see phase diagrams under other back-pressures in Supplemental Material Fig. S1 available in the Supplemental Materials on the ASME Digital Collection). Figure 5(*b*) shows our experimental demonstration to adjust the pinch-off behaviors. At the back-pressure of −4.5, we adjust the dwell voltage and the dwell time to make the droplet front pinch-off. Then, we adjust the dwell voltage and the dwell time to change the droplet pinch-off behavior in the order of exit pinching, hybrid pinching, and middle pinching. In Fig. 5(*a*), the pentagrams are the parameters for demonstrating different pinch-off behaviors, and the Roman numerals show the parameter changing order. The corresponding pinch-off behavior images are shown in Fig. 5(*b*). This application demonstrates the feasibility of adjusting droplet pinch-off behaviors and the significance of identifying droplet pinch-off behaviors in IJP.

It should be noticed that the phase diagram can be considered as a reference for pinch-off adjustment rather than ground truth since it is generated using the collected data at our specific setup. Besides, as long as the printing mechanism is the same, based on our experience and previous experimental results, the phase diagram will be similar if we change the nozzles and the printer from the same vendor.

4.5 Analysis of Intrinsic Reward. We provide a detailed explanation of the intrinsic reward and its parameter selection in this section. To select the parameters of the intrinsic reward, a small number (e.g., five) of experiments of MCRAL without intrinsic reward are needed. In a nutshell, the intrinsic reward can provide the agent with additional information when some conditions are satisfied, and these information will help the agent select those more valuable unlabeled images for annotation. Specifically, the intrinsic reward increases the final reward according to Eq. (9) and further increases the target Q-value according to Eq. (6). When performing stochastic gradient descent to train the agent by solving Eq. (3), the online Q-network will tend to update its parameters close to the

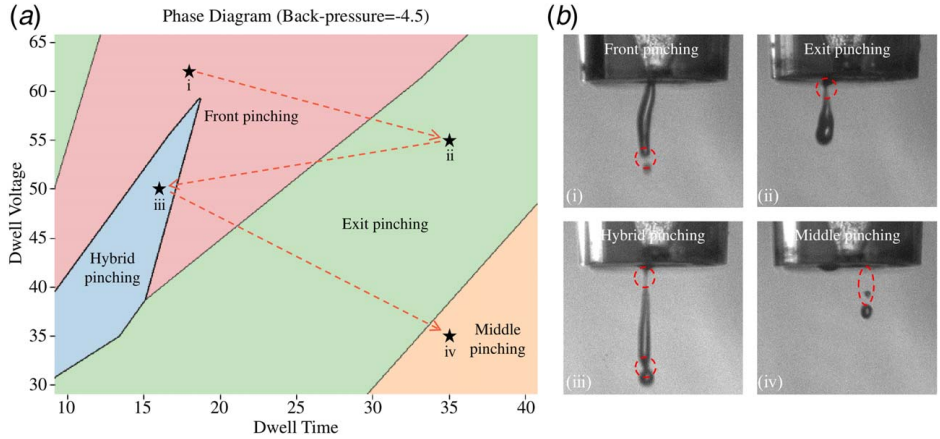

Fig. 5 Demonstration of droplet pinch-off behavior adjustments: (*a*) the phase diagram when the back-pressure equals to −4.5. (See phase diagrams under other back-pressures in Supplemental Material Fig. S1 available in the Supplemental Materials) Different colors represent different droplet pinch-off classes. The pentagrams are the parameters for demonstrating different pinch-off behaviors. The dashed arrows and the Roman numerals show the parameter changing order and (*b*) the corresponding pinch-off behavior images when changing parameters: (i) front pinching, (ii) exit pinching, (iii) hybrid pinching, and (iv) middle pinching. The dashed circles mark the pinch-off positions.

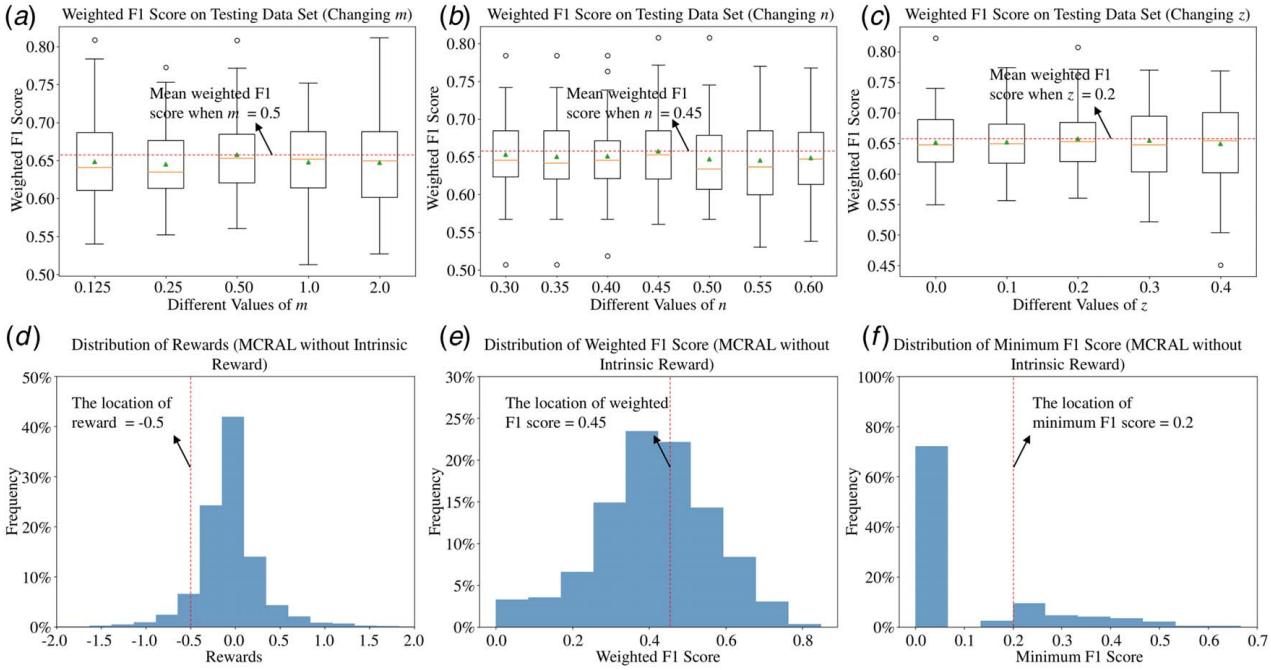

Fig. 6 The sensitivity study of *m*, *n*, and *z* of the intrinsic reward: (*a*)–(*c*) the boxplots of the weighted F1 score of MCRAL with an intrinsic reward on testing data set when changing *m*, *n*, and *z*: (*d*)–(*f*) the distribution of rewards, the weighted F1 score on the validation data set, and the minimum F1 score among all the classes on the validation data set after each query iteration when conducting MCRAL without intrinsic reward.

target Q-value. If the transitions (i.e., S_t, A_t, R_t, S_{t+1}) including the intrinsic reward are sampled during the training, then the agent will tend to update its parameters to take those actions that can generate larger Q-values due to the help of the intrinsic reward.

A sensitivity study is further conducted for the justifications of selecting the parameters of the intrinsic reward (i.e., m, n, and z), as shown in Fig. 6. Each column is the sensitivity study for one parameter. First, we study the impact of the amount of the intrinsic reward, namely m. We fix $n = 0.45$ and $z = 0.2$, and set m equals to 0.125, 0.25, 0.5, 1, and 2, which cover the range of the rewards. The corresponding classification results are shown in Fig. 6(*a*). The triangles and the short lines in the boxplots represent the average weighted F1 scores and the median scores, respectively. The horizontal dashed line is the average weighted F1 score when $m = 0.5$. We can observe that $m = 0.5$ results in the best classification performance. It can be explained with Fig. 6(*d*), which shows the distribution of the reward that the agent received after each query iteration when performing MCRAL without intrinsic reward. In Fig. 6(*d*), we can observe that most rewards fall into the range of −1.5 to 1.5 and they are all extrinsic rewards according to Eq. (9). $m = 0.5$ can offset the majority of negative extrinsic rewards (see the location of −0.5 represented by the vertical dashed line) and can be comparable with most positive extrinsic rewards. In other words, the intrinsic reward can play a role in

Eq. (9), which provides a positive feedback to the agent when the predefined conditions are satisfied, even if the selected unlabeled image leads to the increase of the cross-entropy.

Next, we study n by fixing $m = 0.5$ and $z = 0.2$, and vary n from 0.3 to 0.6 with an interval of 0.05. n is the threshold of the average weighted F1 score which represents the overall classification performance of the classifier, and these numbers are the potential targets. The corresponding classification results are shown in Fig. 6(*b*). It can be seen that $n = 0.45$ (see the horizontal dashed line) results in the best classification performance. This result can be explained in Fig. 6(*e*), which shows the distribution of the weighted F1 score of the validation data set after each query iteration when performing MCRAL without intrinsic reward. We can observe that in Fig. 6(*e*), an appropriate amount of the weighted F1 score can achieve the threshold if $n = 0.45$ (represented by the vertical dashed line). In other words, receiving the intrinsic reward is not too difficult or too easy for the agent, which assures no abuse of the intrinsic reward.

Finally, we focus on the worst classification performance among all the classes measured by the minimum F1 score and z is the threshold. Similar as the study of m and n, we first fix $m = 0.5$ and $n = 0.45$, respectively, then set z to be 0, 0.1, 0.2, 0.3, and 0.4. Figure 6(*c*) shows the corresponding classification results. It can be seen clearly that $z = 0.2$ leads to the best classification performance. Figure 6(*f*) is responsible for the interpretation of this result and it shows the distribution of the minimum F1 score among all the classes on the validation data set after each query iteration when performing MCRAL without intrinsic reward. The vertical dashed line represents the location of $z = 0.2$. Similar as the previous discussion, this value leads to a proper number of intrinsic reward being assigned to the agent, which makes the intrinsic reward play a role but not overwhelm the extrinsic reward.

5 Conclusion

IJP is a promising AM technique that is highly dynamic and the droplet pinch-off behaviors heavily affect the quality of the printed products. There is a need to identify the classes of pinch-off behaviors from the droplet pinch-off images. However, the labeling of these images can be burdensome if done manually. In this paper, we use GCN to extract features from the images. Furthermore, we propose an MCRAL framework with a unique intrinsic reward to train a multiclass classifier for the droplet pinch-off behaviors identification, which saves the annotation efforts. The classification results show the ability of MCRAL to achieve a relatively good classification performance with limited annotating effort. Based on the trained MCRAL, we further link the pinch-off behaviors with the inkjet printing process parameters and demonstrate the pinch-off behavior adjustments during the inkjet printing guided by MCRAL.

There are several directions that we will pursue in the future. First, we will build the process-quality model between the process parameters, the droplet pinch-off behaviors, and the morphology of IJP printed products. Second, we will perform the process control by building controllers based on the process-quality model.

Acknowledgment

This work is partially supported by the NSF Grant No. OAC-2230025, FM-2134409 and CMMI-1846863.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Moya, A., Gabriel, G., Villa, R., and del Campo, F. J., 2017, "Inkjet-Printed Electrochemical Sensors," Curr. Opin. Electrochem., **3**(1), pp. 29–39.
[2] O'Donnell, J., Kim, M., and Yoon, H.-S., 2017, "A Review on Electromechanical Devices Fabricated by Additive Manufacturing," ASME J. Manuf. Sci. Eng., **139**(1), p. 010801.
[3] Joshi, K., Velasco, V., and Esfandyarpour, R., 2020, "A Low-Cost, Disposable and Portable Inkjet-Printed Biochip for the Developing World," Sensors, **20**(12), p. 3593.
[4] Xu, T., Kincaid, H., Atala, A., and Yoo, J. J., 2008, "High-Throughput Production of Single-Cell Microparticles Using an Inkjet Printing Technology," ASME J. Manuf. Sci. Eng., **130**(2), p. 021017.
[5] Segura, L. J., Wang, T., Zhou, C., and Sun, H., 2021, "Online Droplet Anomaly Detection From Streaming Videos in Inkjet Printing," Addit. Manuf., **38**, p. 101835.
[6] Hoath, S. D., 2016, *Fundamentals of Inkjet Printing: The Science of Inkjet and Droplets*, John Wiley & Sons, New York.
[7] Wang, T., Kwok, T.-H., and Zhou, C., 2017, "In-Situ Droplet Inspection and Control System for Liquid Metal Jet 3D Printing Process," Procedia Manuf., **10**, pp. 968–981.
[8] Xu, C., Zhang, Z., Huang, Y., and Xu, H., 2019, "Phase Diagram of Pinch-Off Behaviors During Drop-on-Demand Inkjetting of Alginate Solutions," ASME J. Manuf. Sci. Eng., **141**(9), p. 091013.
[9] Thiévenaz, V., and Sauret, A., 2022, "The Onset of Heterogeneity in the Pinch-Off of Suspension Drops," Proc. Natl. Acad. Sci. U. S. A., **119**(13), p. e2120893119.
[10] Zhu, P., and Wang, L., 2019, "Droplet Pinch-Off With Pressure Fluctuations," Chem. Eng. Sci., **196**, pp. 333–343.
[11] Roché, M., Aytouna, M., Bonn, D., and Kellay, H., 2009, "Effect of Surface Tension Variations on the Pinch-Off Behavior of Small Fluid Drops in the Presence of Surfactants," Phys. Rev. Lett., **103**(26), p. 264501.
[12] Zhang, S., Tong, H., Xu, J., and Maciejewski, R., 2018, "Graph Convolutional Networks: Algorithms, Applications and Open Challenges," Computational Data and Social Networks: Seventh International Conference (CSoNet 2018), Shanghai, China, Dec. 18–20, pp. 79–91.
[13] Aggarwal, C. C., Kong, X., Gu, Q., Han, J., and Yu, P. S., 2014, "Active Learning: A Survey," *Data Classification*, C. C. Aggarwal, ed., Chapman and Hall/CRC, New York, pp. 599–634.
[14] Wang, J., Yan, Y., Zhang, Y., Cao, G., Yang, M., and Ng, M. K., 2020, "Deep Reinforcement Active Learning for Medical Image Classification," Medical Image Computing and Computer Assisted Intervention (MICCAI 2020): 23rd International Conference, Lima, Peru, Oct. 4–8, pp. 33–42.
[15] Liu, M., Buntine, W., and Haffari, G., 2018, "Learning How to Actively Learn: A Deep Imitation Learning Approach," Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, July 15–20, pp. 1874–1883.
[16] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., and Ostrovski, G., 2015, "Human-Level Control Through Deep Reinforcement Learning," Nature, **518**(7540), pp. 529–533.
[17] Van Hasselt, H., Guez, A., and Silver, D., 2016, "Deep Reinforcement Learning With Double Q-Learning," Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, Phoenix, AZ, Feb. 12–17, pp. 2094–2100.
[18] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N., 2016, "Dueling Network Architectures for Deep Reinforcement Learning," Proceedings of the 33rd International Conference on International Conference on Machine Learning (ICML), Vol. 48, New York, NY, June 19–24, pp. 1995–2003.
[19] Schaul, T., Quan, J., Antonoglou, I., and Silver, D., 2015, "Prioritized Experience Replay," preprint arXiv:1511.05952.
[20] Ebert, S., Fritz, M., and Schiele, B., 2012, "RALF A Reinforced Active Learning Formulation for Object Class Recognition," 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, June 16–21, pp. 3626–3633.
[21] Huang, J., Segura, L. J., Wang, T., Zhao, G., Sun, H., and Zhou, C., 2020, "Unsupervised Learning for the Droplet Evolution Prediction and Process Dynamics Understanding in Inkjet Printing," Addit. Manuf., **35**, p. 101197.
[22] Jiao, T., Lian, Q., Zhao, T., and Wang, H., 2021, "Influence of Ink Properties and Voltage Parameters on Piezoelectric Inkjet Droplet Formation," Appl. Phys. A, **127**(1), pp. 1–9.
[23] Tofan, T., Kruggel-Emden, H., Turla, V., and Jasevičius, R., 2021, "Numerical Modeling of the Motion and Interaction of a Droplet of an Inkjet Printing Process With a Flat Surface," Appl. Sci., **11**(2), p. 527.
[24] Huang, J., Wang, T., Segura, L. J., Joshi, G. S., Sun, H., and Zhou, C., 2020, "Spatiotemporal Fusion Network for the Droplet Behavior Recognition in Inkjet Printing," International Manufacturing Science and Engineering Conference, Vol. 84256, Sept. 3, Paper No. V001T001A038.
[25] Wu, D., and Xu, C., 2018, "Predictive Modeling of Droplet Formation Processes in Inkjet-Based Bioprinting," ASME J. Manuf. Sci. Eng., **140**(10), p. 101007.
[26] Mohammadi, K., Movahhedy, M. R., and Khodaygan, S., 2019, "A Multiphysics Model for Analysis of Droplet Formation in Electrohydrodynamic 3D Printing Process," J. Aerosol Sci., **135**, pp. 72–85.
[27] Zhang, L., Zhu, Y., and Cheng, X., 2017, "Numerical Investigation of Multi-Droplets Deposited Lines Morphology With a Multiple-Relaxation-Time Lattice Boltzmann Model," Chem. Eng. Sci., **171**, pp. 534–544.
[28] Andalib, S., Taira, K., and Kavehpour, H. P., 2021, "Data-Driven Time-Dependent State Estimation for Interfacial Fluid Mechanics in Evaporating Droplets," Sci. Rep., **11**(1), p. 13579.

[29] Wang, J., and Chiu, G. T.-C., 2020, "Data-Driven Drop Formation Modeling in Nanoliter Drop-on-Demand Inkjet Printing," Dynamic Systems and Control Conference, Vol. 84287, Virtual, Oct. 5–7, Paper No. V002T028A002.
[30] Li, Z., Lee, J., Yao, F., and Sun, H., 2021, "Quantifying the CVD-Grown Two-Dimensional Materials Via Image Clustering," Nanoscale, **13**(36), pp. 15324–15333.
[31] Rawat, W., and Wang, Z., 2017, "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review," Neural Comput., **29**(9), pp. 2352–2449.
[32] Zhang, S., Tong, H., Xu, J., and Maciejewski, R., 2019, "Graph Convolutional Networks: A Comprehensive Review," Comput. Soc. Netw., **6**(1), pp. 1–23.
[33] Hong, D., Gao, L., Yao, J., Zhang, B., Plaza, A., and Chanussot, J., 2020, "Graph Convolutional Networks for Hyperspectral Image Classification," IEEE Trans. Geosci. Remote Sens., **59**(7), pp. 5966–5978.
[34] Mou, L., Lu, X., Li, X., and Zhu, X. X., 2020, "Nonlocal Graph Convolutional Networks for Hyperspectral Image Classification," IEEE Trans. Geosci. Remote Sens., **58**(12), pp. 8246–8257.
[35] Hemmer, P., Kühl, N., and Schöffer, J., 2022, "DEAL: Deep Evidential Active Learning for Image Classification," *Deep Learning Applications*, M. A. Wani, B. Raj, F. Luo, and D. Dou, eds., Vol. 3, Springer Nature, Singapore, pp. 171–192.
[36] Chang, J., Kim, J., Zhang, B.-T., Pitt, M. A., and Myung, J. I., 2021, "Data-Driven Experimental Design and Model Development Using Gaussian Process With Active Learning," Cogn. Psychol., **125**, p. 101360.
[37] Yue, X., Wen, Y., Hunt, J. H., and Shi, J., 2020, "Active Learning for Gaussian Process Considering Uncertainties With Application to Shape Control of Composite Fuselage," IEEE Trans. Autom. Sci. Eng., **18**(1), pp. 36–46.
[38] Lee, C., Wang, X., Wu, J., and Yue, X., 2022, "Failure-Averse Active Learning for Physics-Constrained Systems," IEEE Trans. Autom. Sci. Eng., pp. 1–12.
[39] Liu, Z., Wang, J., Gong, S., Lu, H., and Tao, D., 2019, "Deep Reinforcement Active Learning for Human-in-the-Loop Person Re-Identification," Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), Seoul, Korea, Oct. 27–Nov. 2, pp. 6122–6131.
[40] Casanova, A., Pinheiro, P. O., Rostamzadeh, N., and Pal, C. J., 2020, "Reinforced Active Learning for Image Segmentation," International Conference on Learning Representations (ICLR), Virtual, Apr. 26–May 1.
[41] Wu, T., and Ortiz, J., 2021, "RLAD: Time Series Anomaly Detection through Reinforcement Learning and Active Learning," preprint arXiv:2104.00543.
[42] Aubret, A., Matignon, L., and Hassas, S., 2019, "A Survey on Intrinsic Motivation in Reinforcement Learning," preprint arXiv:1908.06976.
[43] Bougie, N., and Ichise, R., 2020, "Skill-Based Curiosity for Intrinsically Motivated Reinforcement Learning," Mach. Learn., **109**(3), pp. 493–512.
[44] Baranes, A., and Oudeyer, P.-Y., 2013, "Active Learning of Inverse Models With Intrinsically Motivated Goal Exploration in Robots," Rob. Auton. Syst., **61**(1), pp. 49–73.
[45] Kipf, T. N., and Welling, M., 2016, "Semi-Supervised Classification With Graph Convolutional Networks," Fifth International Conference on Learning Representations (ICLR), Toulon, France, Apr. 24–26.
[46] Jain, A. K., Mao, J., and Mohiuddin, K. M., 1996, "Artificial Neural Networks: A Tutorial," Computer, **29**(3), pp. 31–44.
[47] Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., and Tian, Q., 2019, "Actional-Structural Graph Convolutional Networks for Skeleton-Based Action Recognition," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, June 15–20, pp. 3595–3603.
[48] Zhang, H., Zou, J., and Zhang, L., 2022, "EMS-GCN: An End-to-End Mixhop Superpixel-Based Graph Convolutional Network for Hyperspectral Image Classification," IEEE Trans. Geosci. Remote Sens., **60**, pp. 1–16.
[49] Li, C., Meng, Y., Chan, S. H., and Chen, Y.-T., 2020, "Learning 3D-Aware Egocentric Spatial–Temporal Interaction Via Graph Convolutional Networks," 2020 IEEE International Conference on Robotics and Automation (ICRA), Virtual, May 31–June 15, pp. 8418–8424.
[50] Danka, T., and Horvath, P., 2018, "modAL: A Modular Active Learning Framework for Python," preprint arXiv:1805.00979.
[51] Weng, J., Chen, H., Yan, D., You, K., Duburcq, A., Zhang, M., Su, Y., Su, H., and Zhu, J., 2022, "Tianshou: A Highly Modularized Deep Reinforcement Learning Library," J. Mach. Learn. Res., **23**(267), pp. 1–6.