A Deep Learning Quality Control Loop of the Extrusion-based Bioprinting Process

Amedeo Franco Bonatti¹, Giovanni Vozzi¹, Chee Kai Chua², Carmelo De Maria¹*  

¹Department of Information Engineering and Research Center “Enrico Piaggio,” University of Pisa, Pisa, Italy  
²Engineering Product Development Pillar, Singapore University of Technology and Design, Singapore

Abstract: Extrusion-based bioprinting (EBB) represents one of the most used deposition technologies in the field of bioprinting, thanks to key advantages such as the easy-to-use hardware and the wide variety of materials that can be successfully printed. In recent years, research efforts have been focused on implementing a quality control loop for EBB, which can reduce the trial-and-error process necessary to optimize the printing parameters for a specific ink, standardize the results of a print across multiple laboratories, and so accelerate the translation of extrusion bioprinted products to more impactful clinical applications. Due to its capacity to acquire relevant features from a training dataset and generalize to unseen data, machine learning (ML) is currently being studied in literature as a relevant enabling technology for quality control in EBB. In this context, we propose a robust, deep learning-based control loop to automatically optimize the printing parameters and monitor the printing process online. We collected a comprehensive dataset of EBB prints by recording the process with a high-resolution webcam. To model multiple printing scenarios, each video represents a combination of multiple parameters, including printing set-up (either mechanical or pneumatic extrusion), material color, layer height, and infill density. After pre-processing, the collected dataset was used to thoroughly train and evaluate an ad hoc defined convolutional neural network by controlling over-fitting and optimizing the prediction time of the network. Finally, the ML model was used in a control loop to: (i) monitor the printing process and detect if a print with an error could be stopped before completion to save material and time and (ii) automatically optimize the printing parameters by combining the ML model with a previously published mathematical model of the EBB process. Altogether, we demonstrated for the first time how ML techniques can be used to automatize the EBB process, paving the way for a total quality control loop of the printing process.

Keywords: Extrusion-based bioprinting; Quality control; Convolutional neuronal network; Automatic parameter optimization

*Correspondence to: Carmelo De Maria, Department of Information Engineering and Research Center “Enrico Piaggio,” University of Pisa, Pisa, Italy; carmelo.demaria@unipi.it

Received: August 26, 2022; Accepted: September 6, 2022; Published Online: October 11, 2022

(Citation: Bonatti AF, Vozzi G, Chua CK, et al., 2022, A Deep Learning Quality Control Loop of the Extrusion-Based Bioprinting Process. Int J Bioprint, 8(4):620. http://doi.org/10.18063/ijb.v8i4.620)
aspects of the EBB process, including material, scaffold geometry, and printing apparatus that determines a priori if the printing process will be successful or not)\(^{[5-9]}\). However, there is often a delicate balance between these different requirements, which results in careful and time-consuming optimization of the material properties, printing parameters, and scaffold geometry to obtain good print quality results. Furthermore, since no standardized method is currently available for the printing parameters optimization, the results from this trial-and-error process may not be reproducible across labs. Altogether, these limitations pose a significant obstacle to the translation of a promising ink/bioprinted product to more impactful clinical applications, as it is more difficult to comply to relevant healthcare-related standards\(^{[10]}\).

In the last few years, artificial intelligence (AI) technologies have been proposed to improve the control and automatization of the complex EBB process\(^{[11-14]}\). In this regard, Fu et al. optimized the printing settings in EBB using the support vector machine (SVM) machine learning (ML) model. In their research, the authors established an initial set of printing settings to be adjusted, including cartridge temperature, layer height, and needle gauge for a Pluronic F127-based biomaterial ink. The authors printed three-layer grid-like designs using various combinations of these parameters, and they assessed the faithfulness by measuring the line width and comparing it with the theoretical one. A 3D map was generated using the SVM model, indicating the parameter combination that could yield higher quality prints within a given probability threshold\(^{[15]}\). In another recent work, Rubero et al. proposed an iterative method based on Bayesian optimization for the optimization of EBB parameters (e.g., biomaterial ink composition, reservoir and bed temperatures, extrusion pressure, and printing speed). Briefly, a set of lattice structures with random parameters were printed to begin the optimization process. These first prints were visually evaluated to compute a quality index and employed to create a probabilistic model based on a Gaussian process. A new set of printing parameters was proposed by the model and the experimenter used these parameters to create a fresh batch of prints and corresponding scores. The process was repeated until convergence\(^{[16]}\).

The use of more sophisticated ML models for anomaly identification and localization was studied by Jin et al. Specifically, the authors printed several infill patterns, such as grid, rectilinear, gyroid, and honeycomb, at various printing speeds while taking top-down pictures of the first layer. The photos were divided into several categories, such as good prints and prints with errors. To automatically classify the prints, various ML models, including SVM and a deep learning (DL) convolutional neural network (CNN), were tested. The CNN-based strategy was shown to be the best fit for anomaly detection on the provided dataset\(^{[17]}\).

Despite the good results of these few preliminary studies, much research is still needed for the implementation of a robust, AI-based quality control system of the EBB process. First, this system should be able to analyze the printing outcome and change the printing parameters without any manual involvement to have a completely autonomous control loop. Furthermore, the quality assessment should focus on the full, 3D scaffold shape, and not only on a few, initial layers, which are not representative of the final shape. Finally, the system should also be able not only to optimize the printing parameters but also monitor the printing session for quality assessment and potential on-the-fly correction of working parameters.

Considering these challenges, we present a novel approach for parameter optimization and in-process monitoring of the EBB process using a combination of: (i) a robust DL model (based on an ad hoc optimized neural network) which can classify the print outcome into one of three classes (i.e., good, under, and over extrusion) and (ii) a mathematical model (already published in Bonatti et al.\(^{[18]}\)) to automatically tune the printing parameters toward the optimal combination. Briefly, we built a comprehensive dataset by taking videos of the EBB process from a frontal view while printing multilayer scaffold shapes. To model multiple printing outcomes, for each print, a different combination of relevant parameters (e.g., layer height, flow, infill, color of the material, and printing set-up) was tested. The dataset is publicly hosted in Zenodo (https://doi.org/10.5281/zenodo.7024007). After frame extraction and pre-processing, the dataset was used to train and comprehensively evaluate a CNN architecture with high classification accuracy. Finally, we demonstrated how the classification model could be used in a feedback loop to monitor the printing outcome and automatically optimize the printing parameters through a series of consecutive prints.

The paper is organized as follows: given the limited amount of literature on AI-based quality control for EBB, in Section 2, we give the reader both a brief introduction on ML and DL, as well as an analysis on the current literature on the use of ML for the broader field of additive manufacturing. Through this section, the reader will obtain a more comprehensive overview of recent techniques, algorithms, and applications, which serves as a basis for the work described in the remaining sections. In Section 3, we describe the main methods used to build the dataset and train and evaluate the DL model, as well as how this can be used in a feedback loop for online monitoring and automatic parameter optimization. In Section 4, we detail and discuss the results achieved during the experiments, while Section 5 will be focused on the conclusions and future developments of the work.
2. ML techniques in additive manufacturing

ML refers to a group of AI techniques that enable automatic learning from data to make decisions or predictions without being explicitly programmed to do so. Traditional ML algorithms, such as SVM, random forest, and logistic regression, are limited by the high dimensionality of input data (e.g., images), the handling of time series (e.g., video), and the large amount of data to process. In general, DL algorithms can be used to implement a CNN to extract relevant features from their inputs (e.g., edges in the input image for the top layers of the CNN). On the other hand, pooling layers are used to downsample the output of convolutional layers and so summarize the features learned from them.

To overcome these limitations, researchers have moved toward DL and deep neural networks (DNNs), which have demonstrated to scale much better with the increase of data size. In general, DL algorithms can be classified into three main groups depending on the type of data available. In supervised learning, the network is used as a binary or multiclass classifier using labeled data instances. Semi-supervised models use a small amount of labeled data together with a larger amount of unlabeled data, while in unsupervised learning, no labeled data are available. At its core, DNNs use a composition of linear and non-linear functions to learn an expressive representation of the data. The term “deep” refers to stacking multiple layers (i.e., a set of neurons) to obtain more complex function approximators. The type of layer to be used in a neural network depends on the type of data and processing of interest. For example, the basic architecture of a CNN, designed to operate on data in array format (e.g., a stack of three 2D arrays corresponding to the pixel values of a color image), is given by the repetition of convolutional and pooling layers. Each convolutional layer implements multiple filters, whose weights are optimized during training to extract relevant features from their inputs (e.g., edges in the input image for the top layers of the CNN). On the other hand, pooling layers are used to downsample the output of the convolutional layers and so summarize the features learned from them.

The properties of DNNs make them good candidates for multiple applications in the field of AM, including: (i) designing new composite materials and topologies (e.g., given a target Young’s modulus, optimize the structure to achieve it considering the constraints of AM), (ii) optimizing the process parameters, and (iii) monitoring the printing process to detect defects (e.g., cracks, delamination, and porosity). Regarding these last two points, several works have focused on applying DL to fused deposition modeling (FDM), inkjet, and powder sintering/melting processes.

For example, Jin et al. used a CNN to implement a real-time feedback loop on solid (no infill) FDM-printed layer. The authors identified three main classes (i.e., under-extrusion, over-extrusion, and good quality) and built a dataset of images using a top view of the printing process. The collected data were used to test and validate the DL model, reaching a high classification accuracy (around 98%). In another work, Tonnaer et al. tackled the problem of detecting anomalies on the surface of FDM-printed parts using a semi-supervised approach. The authors employed a variational autoencoder, a type of deep architecture which, when trained with images from non-faulty parts, can learn their probability distribution. When fed with images from error containing parts, the model can then assign a value to those images representing the probability that they come from the same distribution of non-faulty images. As a result, by setting a proper threshold, the model can effectively distinguish between good and erroneous prints. The work by Zhang et al. highlighted how multiple information coming from different sensors can be integrated together for process monitoring. In particular, the authors proposed a DL model to understand the tensile properties of FDM-printed parts. They added a set of sensors to a commercial printer, including infrared sensors to measure the temperature of the printed layer and accelerometers to quantify vibrations. The layer-wise output of these sensors was fed to a long short-term memory (LSTM) network, whose output, together with other relevant information (e.g., extruder temperature and printing speed) was used to predict the tensile strength of the printed part.

In the field of laser-based technologies, Zhang et al. proposed a method for in-process monitoring of direct laser deposition. Briefly, a high-resolution camera was used to record the formation of a single line of material. After stitching together multiple successive images, the collected data were used to train and evaluate a CNN model to both detect the presence of porosity (with a 91% accuracy) and predict the overall volume porosity (with a root mean squared error of 1.3%). Finally, Wang et al. implemented a closed-loop control to stabilize the printing process in inkjet printing. In particular, the authors used a vision-based system to monitor the droplet formation process. Feature extraction was performed on the images by: (i) cropping around a region of interest (ROI), (ii) binarizing the image, and (iii) extracting connected components information, such as area and position. These features were then fed to a neural network to detect abnormal droplets and compensate them using a feedback loop with proportional integral derivative (PID) system controlling the operational voltage.

3. Materials and methods

3.1. Material preparation

Poloxamer 407 (commercial name Pluronic F-127) was used for all printing experiments. A solution of Pluronic acid F-127 (Sigma-Aldrich) at 25% w/v was prepared by gradually dissolving the Pluronic powder in deionized water at 90°C through magnetic stirring. After complete dissolution
of the powder, the solution was left at rest at 4°C overnight before printing\(^7\). To increase variability in the dataset, the Pluronic solution was used both as is (transparent) as well as colored using a red and a blue colorant.

3.2. Software implementation and hardware for training

All code to process data, train and evaluate the model, and implement the control loop was written in Python. The data processing was performed using the libraries: Pandas\(^{36}\) for data handling, OpenCV\(^{37}\) for image preprocessing, and TensorFlow\(^{38}\) for efficient data loading. The models were defined and trained using the Keras API\(^{39}\) and the scikit-learn library\(^{40}\). All random function calls (i.e., data pre-processing and splitting, training, and evaluation) as well as the global seed value for the used modules were initialized using the same integer (i.e., 1234) to ensure repeatability of the results across multiple calls. The code is available at https://doi.org/10.5281/zenodo.7024016 to reproduce the training and evaluation results. All data analysis, statistical tests, and graphing were performed using GraphPad prism (version 8.0.2).

All models were trained on a dedicated computer with an Intel Core i9 processor (running at 3.3–4.6 GHz), 32 GB of RAM and equipped with a GeForce RTX 3090 graphics card. A laptop with an Intel Core i7 processor (running at 1.8 GHz) and 16 Gb of RAM was used to evaluate the prediction speed of the final optimized model using CPU only.

3.3. Dataset definition

3.3.1. Dataset design and collection

In this work, we examined two distinct extrusion modalities: pneumatic assisted and piston actuated. These two methods were chosen to expand the variety of the printing scenarios and hence enhance the trained model capacity for generalization. In particular, we employed an Alleli 1 bioprinter for the pneumatically assisted mechanism and a purposely built bioprinter for the piston-actuated one\(^{41}\).

To train and evaluate the DL model, we built a dataset of videos using a high-definition webcam (Logitech C920 at 1920 × 1080 resolution). The webcam was positioned in front of the printers at varying distances (Logitech C920 at 1920 × 1080 resolution). The webcam was positioned in front of the printers at varying distances (Logitech C920 at 1920 × 1080 resolution). The webinar was positioned in front of the printers at varying distances and recorded the whole printing process from this view. Parallelepiped-shaped scaffolds of 10 mm × 10 mm side and 5 mm height were printed at ambient temperature.

After printing, each video was assigned a label depending on the final shape of the scaffold: “Ok,” under-extrusion (“under e”) (i.e., not enough material has been extruded) and over-extrusion (“over e”) (i.e., an excess of material has been deposited). To model multiple printing scenarios, each video corresponded to a distinct set of parameters. A list of varied parameters is shown in Table 1. The dataset is available at https://doi.org/10.5281/zenodo.7024007.

In Table 1, the layer height (LH) represents the distance between the needle and the previously printed layer (or printing plate for the first layer) and is referred to as a percentage of the needle diameter \(D_N\). The extrusion multiplier (EM, also called “flow”) represents a modifier of the amount of material that is being extruded. This parameter is computed as a percentage of a standard flow for extruding cylindrical strands on free air with a diameter equal to the nozzle diameter \(D_N\)\(^7\).

3.3.2. Data partitioning and image pre-processing

The collected videos were first randomly down sampled to the lower representation class to ensure class balance in the dataset. A total of 345 videos (115 per each class) were employed for model training and evaluation. The videos were randomly partitioned into training and testing sets using, respectively, 90% and 10% splits (splits stratified over the video class).

Each video was converted to a frame sequence by sampling one frame per second. Since it was observed that the printing error was more apparent at the end of the video, only the last 90 frames from each video were used (totaling around 30K individual frames). Then, each frame was passed through a pre-processing step to reduce the problem complexity. The main steps of the pipeline are summarized in Figure 1A. Briefly, each frame was converted from red green blue (RGB) to hue saturation value (HSV) color space to extract the value (V) channel. Contrast limited adaptive histogram equalization (CLAHE) was applied to each grayscale ROI to increase the image contrast. Finally, each image was normalized from the original range 0 – 255 to the range 0 – 1 to avoid numerical under- or over-flow during training.

During training only, the frames were augmented by applying a set of random operations that simulate other variability not accounted in the dataset generation, as shown in Figure 1B. A summary of all applied transformation is found in Table 2, alongside the values for each operation chosen after preliminary experimentations. The augmentation transformations were applied in sequence and online, so that at each training epoch the model was presented with a slightly diverse set of images.

3.4. Model optimization procedure

The two main requirements that guided the model architecture selection and optimization were: (i) good generalization
performance by controlling over-fitting and (ii) fast prediction time by reducing the model number of parameters.

Taking these requirements into account, we designed an *ad hoc* CNN architecture, which is shown in Figure 2A. It consists of repeating multiple convolutional blocks ("conv block" for short), each having a different structure depending on the type chosen (Figure 2B). For the "simple" case, the "conv block" is given by: (i) a 2D

\[
\text{(i) } \text{a } 2D
\]

\[
\text{Table 1. Summary of the main parameters varied to create the video dataset}
\]

| Printer configuration | Parameter name                               | Value                           |
|-----------------------|----------------------------------------------|---------------------------------|
| Piston actuated       | Material color                               | Transparent                     |
|                       | LH (relative to the needle diameter)         | 0.5 – 0.7                       |
|                       | EM                                           | 0.7 – 1 – 1.3                   |
|                       | \(S_{\text{fill}}\)                          | 50% – 100%                      |
|                       | Background                                   | Uniform black background/no background |
|                       | Needle geometry                              | Cylindrical                     |
|                       | Needle diameter                              | 0.41 mm                         |
| Pneumatic assisted    | Material color                               | Transparent – Red – Blue        |
|                       | LH                                           | 0.3 – 0.5 – 0.7                 |
|                       | Pressure                                     | \(\approx 124 \text{ kPa} – \approx 138 \text{ kPa} – \approx 152 \text{ kPa}\) |
|                       | \(S_{\text{fill}}\)                          | 30% – 50% – 70% – 100%          |
|                       | Background                                   | No background                   |
|                       | Needle geometry                              | Cylindrical                     |
|                       | Needle diameter                              | 0.23 mm – 0.41 mm               |

LH, layer height; EM, extrusion multiplier; \(S_{\text{fill}}\), infill percentage

\[
\text{Table 2. List of the random image augmentation transformations applied (in the presented order) during training of all models}
\]

| Transformation         | Intensity | Description                                                                 |
|------------------------|-----------|-----------------------------------------------------------------------------|
| Random flip            | -         | Horizontal flip                                                             |
| Random translation     | 20%       | Translates the image horizontally and vertically using a random value in range \([-\text{intensity}, +\text{intensity}]\). The percentage is referred to the image height for vertical translations, and image width for horizontal ones |
| Random zoom            | 10%       | Zoom the image using a random value in range \([-\text{intensity}, +\text{intensity}]\) |
| Random contrast        | 0.2       | Applies contrast adjustment by picking a value in range \([-\text{intensity}, +\text{intensity}]\). Knowing the mean of the image, for each pixel \(x\) the following function is applied: \((x - \text{mean}) \times \text{contrast factor} + \text{mean})
Deep learning for EBB control

A convolution layer to extract features from the block input (the image for the top-level block and the feature map of the previous block for the others), (ii) a rectified linear unit (ReLU) activation function which introduces non-linearity to the model, and (iii) a max pooling layer to downsample the feature map by a factor of 2. The “vgg” architecture is like the “simple” one, but instead of having a single convolution-activation layer, it has two before max pooling (as in the original VGG architecture\cite{43}). Finally, the “conv block” for the “resnet” model implements skip connections as shown in He et al.\cite{44}.

Considering the size of the collected dataset (around 30K unique frames), we used 2D global average pooling (i.e., layer that computes the global mean of each input feature map) instead of a flatten followed by a dense layer to reduce the overall number of parameters and so limit overfitting. The output of the average pooling layer is fed to a final dense layer (with “softmax” activation) which performs the multiclass classification.

For final model selection, we considered two relevant parameters that may influence the performance of the CNN: (i) depth, that is, the number of repeating “conv blocks” before the average pooling, which influences the total number of parameters and the complexity of the extracted features and (ii) the “conv block” type. A list of chosen values for these parameters is reported in Table 3, alongside other relevant hyperparameters that were not changed during model selection.

To avoid over-fitting the parameters to a specific validation set, we performed a 5-fold cross validation (stratified based on the class) using the training videos described in Section 3.3.2. Early stopping by monitoring the validation loss with a patience of 20 epochs (i.e., stop training if the validation loss reaches a minimum and does not decrease significantly from that value after 20 epochs) was used to avoid over-fitting and limit the computational time of the cross-validation. The model selection was performed by looking at both the validation accuracy and loss (mean over the 5-folds).

A two-way analysis of variance (two-way ANOVA; $\alpha = 0.95$) was performed to investigate any statistically significant effect on these two metrics of the depth and “conv block” type.

After choosing a final model set-up, the whole training set was used to train the DL model from scratch. As done during cross-validation, early stopping was implemented by monitoring the loss on the test set (defined in Section 3.3.2) with a patience of 20 epochs. The resulting model was used to predict over the frames from the test videos by computing the confusion matrix as well as the overall accuracy and per-class precision and recall Equation I:

$$\text{Accuracy} = \frac{\text{Number of samples correctly classified}}{\text{total number of samples}} \quad (1)$$
Precision = \frac{TP_i}{TP_i + FP_i} \hspace{1cm} \text{(II)}

Recall_i = \frac{TP_i}{TP_i + FN_i} \hspace{1cm} \text{(III)}

The subscript \( i \) in the Equations II and III indicates one of the three classes. For example, the precision for the “ok” class is defined as the ratio between the “ok” frames classified as “ok” (true positives, \( TP_i \) in Equation II) over the overall number of frames classified as “ok” (sum of true positives, \( TP_i \), and false positives, \( FP_i \), in Equation III). This last term may include frames that were classified as “ok,” but their “true” value is from another of the two remaining classes.

Finally, although metrics are useful to quantitatively evaluate model performance, it is also important in practice to verify that the model is behaving as expected. To this end, we first tested its performance by taking snapshots of the same print under different conditions, including different zoom and focus levels. For each snapshot, we verified that the model was predicting the image class correctly and consistently. Then, we employed the gradient-weighted class activation mapping (grad-CAM) technique to verify that the model was focusing on the scaffold shape and not predicting based on elements from the background. Briefly, grad-CAM is one of the most popular techniques to explain the results of a CNN. It involves using the feature maps from the last convolutional layer (weighted based on a gradient function) to compute a heatmap in which the higher values correspond to the most important image regions for the prediction\[47\].

### 3.5. In-process monitoring

Having optimized the DL model to classify single frames, the next step of the work was to verify if the classifier could be used to monitor the printing process online. Different strategies can be envisioned for this task; herein, we decided to use a running moving average filter on the per-class predicted probability to reduce as much as possible prediction flickering across a video acquisition, which may be due to environmental effects like illumination changes. After preliminary experimentation (data not reported), we chose a window size of 30 frames for the average filter. At each time step, the overall classification is then given by the class with the highest predicted probability among the three.

To evaluate the in-process monitoring performance, we printed four scaffolds at varying EM (parallelepiped scaffold of 10 mm × 10 mm side, 5 mm height, infill density at 30%, LH at 70%, transparent Pluronic, and using the pneumatic-based bioprinter) and then plotted the filtered per-class probabilities. By analyzing the resulting graphs, we inferred if a print with an error could be stopped before completing to reduce material consumption and decrease the process time.

### 3.6. Automatic parameter optimization

Figure 3A reports the general pipeline for the automatic parameter optimization. As can be seen from the figure, the process begins with a new ink to be printed. The user needs to decide a starting point for the LH (LH\(_i\)) and infill density, and the system will optimize the EM parameter to obtain an “ok” print if possible.

In particular, the parameter optimization system uses the concept of printability window introduced in Bonatti \textit{et al.}\[7\]. An example of such graph is shown in Figure 3B. Briefly, the central zone indicates the combination of EM and LH that allow the formation of a good-quality printed line for the first layer. If EM is too high in respect to LH, the material will accumulate around the needle, resulting in over-extrusion (“EM\(_{\text{max}}\)” in the figure); whereas if the LH is too high with respect to the EM, the deposited line will break-up due to under-extrusion (“EM\(_{\text{min}}\)” in the figure). It is important to stress out that: (i) the window is valid only for the first layer and (ii) that the analysis done through the mathematical model in Bonatti \textit{et al.}\[7\] is dependent on the material properties (e.g., yield stress, 

| Parameter group | Parameter name                        | Values                          |
|-----------------|---------------------------------------|---------------------------------|
| Optimized parameters | Depth                               | 5, 6, 7                         |
|                 | “conv block” type                     | Simple, vgg, resnet              |
| Model architecture | Activation function of hidden layers  | ReLu                            |
|                 | Output layer activation function      | Softmax                         |
|                 | Weight initialization                 | HeUniform\[45\]                  |
|                 | Convolution layer kernel size         | 3×3                              |
| Training        | Loss                                  | Categorical cross-entropy        |
|                 | Optimizer                             | Adam\[46\]                      |
|                 | Learning rate                         | 1e-3                            |
|                 | Batch size                            | 128                             |

### Table 3. Summary of the optimized parameters as well as other relevant hyperparameters for the training process
viscosity, and elastic modulus) which are unknown in our case.

The optimization system starts with the EM suggested by the initial printability window at the given LH \( \text{LH}_i \) (Figure 3B, red dot in the first graph of the “init calibration” step). Furthermore, at this stage, it also computes the difference between the two values for \( EM_{\text{max}} \) and \( EM_{\text{min}} \) evaluated at the specified LH Equation IV:

\[
\Delta = EM_{\text{max}} \left( LH_i \right) - EM_{\text{min}} \left( LH_i \right) \tag{IV}
\]

The \( \Delta \) will be used as an update strength for the EM during all steps in the optimization procedure. After printing, the DL model automatically evaluates the quality using the methods described in the previous sections. If the DL model prediction is an error, we update the printability window by making the first point the border point of one of the two corresponding cases (Figure 3B, green dot in the second graph of the “init calibration”). This is equal to increasing (if the predicted class is “under_e”) or decreasing (if the predicted class is “over_e”) the EM of the previous step by a factor of \( \Delta/2 \). The process is then repeated until a good print is detected by the DL model.

At the end of the “init calibration” step, a new printability window has been defined and will not be updated in the following steps. However, even if the DL model may show a high accuracy in testing, it may fail to detect the correct quality of the print due to prediction uncertainty. To balance this problem, we introduce a second step called the “small perturbations” step in the figure. After detecting a good quality print, we repeat the process by updating the EM by gradually smaller values (e.g., \( \Delta/4 \) and \( \Delta/8 \)) until the DL model predicts “ok” again. The final optimized EM will be given by the mean value between the optimized EM of the “init calibration” procedure and the one from the “small perturbations” step.

To avoid wasting too much material and time, we limit the overall number of updates (sum of the “init calibration” and “small perturbations” steps) using a maximum iteration parameter (“max iters” in Figure 3A), which can be decided by the user (default value equal to 5). To test the overall procedure, we printed a simple scaffold shape of 10 mm sides and 5 mm height using an infill density of 50% and a LH of 0.7. For this final test, we chose to use the transparent Pluronic solution, which was printed with the pneumatic-based extrusion bioprinter.

4. Results and discussion

4.1. Model optimization procedure

Figure 4 reports the main results related to the model selection experiments. As shown in Figure 4A, all tested
models (in terms of depth and “conv block” type) reached high mean accuracies (above 90%) on the 5-fold of the cross-validation procedure. The results from the two-way ANOVA tests showed no statistically significant effects of the two tested parameters on both the validation accuracy and loss ($P > 0.05$ for both cases). As a result, we chose the final model based on the number of parameters (which is an indication of the model complexity). Considering the graph in Figure 4A, we selected the configuration with depth = 6 and a simple “conv block,” representing an intermediate solution between a high (meaning slower computation time) and low complexity (which may not generalize well to new printing scenarios). The optimized DL model showed a fast computing time of around 182 ms to classify 30 frames (average of 10 runs on CPU), which is compatible with the 1 s sampling frequency for the in-process monitoring application.

Figure 4B reports the training loss and accuracy curves for the final model. The model started over-fitting after just 6 epochs, as confirmed by the increase of the test loss curve. This may be due to the fact that, even if a set of augmentation operations had been applied to the dataset, the actual shape of the scaffold for each class did not show a great variability, making the problem easy to learn for the model. The confusion matrix in Figure 4C, computed using the saved model at this step, shows that the DL model can predict correctly over the dataset with high values of the computed metrics, as reported in Table 4.

It is important to note that for the “ok” class the precision is significantly lower than the precision for the other classes. This means that the DL model may

| Group  | Metric | Value |
|--------|--------|-------|
| Overall | Accuracy | 94.3% |
| “ok”   | Precision | 87.2% |
|        | Recall   | 96.5% |
| “over_e” | Precision | 98.3% |
|        | Recall   | 94.5% |
| “under_e” | Precision | 97.6% |
|        | Recall   | 92.2% |
misunderstand some prints as “ok” when their true class is different. This behavior was observed especially for those printing parameters that are close to the optimal and in which the resulting shape is difficult to classify from the front view videos even by the experimenter. This uncertainty was accounted for in the automatic parameter optimization procedure by introducing the “small perturbations” step, as previously described.

Figure 4D and E shows the results related to the robustness evaluation of the DL model. As shown in Figure 4D, the model correctly predicts the class even if there are changes in zoom and focus levels. Furthermore,

Figure 5. Example plots of the prediction probability for each class over time. The vertical red dotted lines represent the print progress after which the model correctly classifies the print.

Figure 6. Example calibration procedure across four prints. For the case of $L_H = 0.7$, from the initial printability window, we have that $\Delta = 0.31$. The vertical red dotted lines represent the print progress after which the model correctly classifies the print.
the grad-CAM analysis shows that the model is not picking up elements of the background for classification. It is also interesting to note that the model focuses more on the top portion of the scaffold to classify error prints, as shown in Figure 4E.

4.2. In-process monitoring

Example plots of the per-class prediction probability over time (smoothed using the rolling average filter with window size equal to 30 frames) are reported in Figure 5 for prints of each class. The X-axis of the plots represents the percentage of print progress over time (with one representing a finished print); note that for all axes, the data before 20% progress are not reported since in those frames, the deposited material could not be seen (due to occlusion with the print support). As can be seen from the figure, the “under_e” is detected as soon as the material becomes visible. This can be explained by the fact that the under-extrusion pattern remains similar throughout a print. On the other hand, the “ok” and “over_e” prints need a more complete print to be safely detected.

Because of this variability, a global stop criterion, which is good for all classes, is difficult to formulate. However, using a safety margin, the print may be stopped at around 80% completion if an error is detected, making the print parameter optimization 20% faster, and avoiding wasting 20% of the printing material.

4.3. Automatic parameter tuning

Figure 6 shows an example of an automatic printing parameter optimization. As can be seen from the figure, only four successive prints (two for the “init calibration” and two for the “small perturbations” step) were necessary to optimize the EM. The two “ok” prints were performed at EM = 1.38 and EM = 1.30, respectively, so the model optimization procedure outputs the best EM as 1.34 (mean value between the two). Note that a total of around 1.2 mL were used during printing, which can be lowered to around 1 mL by stopping the prints at 80% progress as previously described.

5. Conclusions

In this work, we proposed for the 1st time an AI-based quality control loop that can be used to both automatically optimize the printing parameters for a given material and printing set-up, as well as monitor the printing status online with a fast response time. We developed a comprehensive dataset (available at https://doi.org/10.5281/zenodo.7024007) of the EBB process by taking videos of different prints with a combination of multiple parameters, including LH, EM, infill density, extrusion system, and material color.

Regarding the dataset creation, we used Pluronic F-127 since it is well known for its printability. It is important to stress out that the use of other materials has been accounted for by introducing color to the Pluronic solution and converting the RGB frame to grayscale. As a result, since the CNN model is only interested in the appearance of the material and not on its properties (e.g., viscosity and yield stress), the frames for other materials will appear like those for the Pluronic, effectively limiting the number of tests to be performed.

A CNN architecture was trained and comprehensively evaluated on this dataset to obtain a robust model which is: (i) not prone to over-fitting and (ii) fast to predict the classes for a set of video frames. The code for training and evaluation is available at https://doi.org/10.5281/zenodo.7024016 for reproducible results. The optimized DL model showed good classification results and robustness after being evaluated on a separate test split, making it a suitable candidate to be used in the feedback loop. Herein, we proposed a method for the fast optimization of the printing parameters, based on updates on the EM parameter and guided by a previously published mathematical model of the EBB process. We demonstrated that the control loop can automatically optimize the EM value for a specific LH and scaffold infill density in just four steps, with a considerable reduction of time and material waste when compared to manual experimentation. Furthermore, thanks to the fast response of the DL model (even when tested on non-dedicated, non-GPU accelerated hardware), the system can monitor the printing process for all successive prints and can be programmed to modify the EM parameter by small increments to compensate for potential problems in future prints (e.g., change in material properties over time).

Future work will be focused on expanding the dataset with new scenarios, including for example different bioprinters, scaffold dimensions, and shapes. Moreover, readings from other sensors (e.g., syringe and printing plate, extrusion pressure) will be integrated in the loop to enrich the process control and help detect and classify other types of errors, which were not accounted for in this work.

Acknowledgments

All authors acknowledge the support of the Crosslab Additive Manufacturing and the Crosslab Cloud Computing, Big Data & Cybersecurity of the Department of Information Engineering of the University of Pisa. All authors would like to thank Brandon Kaiheng Tay for his help in collecting the dataset.

Funding

The work was supported by: (i) the European Union’s Horizon 2020 research and innovation program under the project GIOTTO: “Giotto: Active ageing and osteoporosis:
The next challenge for smart nanobiomaterials and 3D technologies,” grant agreement no. 814410; (ii) the University of Pisa project PRA mOSAiC: Open Source as key enabling approach for AI in healthcare (PRA_2020_38); (iii) the Italian Ministry of University and Research (MUR) under the PRIN Project VISION “Development and promotion of the levulinic acid and carboxylate platforms by the formulation of novel and advanced PHA-based biomaterials and their exploitation for 3D-printed green electronics applications,” grant agreement no. 2017FWC3WC; and (iv) the Regione Toscana project TRITONE (bando ricerca e salute 2018).

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

Author contributions
Conceptualization: Carmelo De Maria

Formal analysis: Carmelo De Maria

Funding acquisition: Giovanni Vozzi and Chee Kai Chua

Investigation: Amedeo Franco Bonatti

Methodology: Amedeo Franco Bonatti and Carmelo De Maria

Software: Amedeo Franco Bonatti

Supervision: Giovanni Vozzi and Chee Kai Chua

Validation: Amedeo Franco Bonatti

Writing – Original draft: Amedeo Franco Bonatti

Writing – Review and editing: Carmelo De Maria

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Availability of data
The dataset collected in this study is available at: https://doi.org/10.5281/zenodo.7024007. The code to train and evaluate the model is available at: https://doi.org/10.5281/zenodo.7024016.

References
1. Santoni S, Gugliandolo SG, Sponchioni M, et al., 2021, 3D bioprinting: Current status and trends a guide to the literature and industrial practice. Biodes Manuf, 5: 14–42. https://doi.org/10.1007/s42242-021-00165-0

2. Moroni L, Boland T, Burdick JA, et al., 2018, Biofabrication: A guide to technology and terminology. Trends Biotechnol, 36: 384–402.

3. Bonatti AF, Fortunato GM, de Maria C, et al., 2022, Bioprinting technologies: An overview. Bioprinting, 19-49. https://doi.org/10.1016/B978-0-323-85430-6.00006-6

4. Pati F, Jang J, Lee JW, et al., 2015, Extrusion bioprinting. In: Essentials of 3D Biofabrication and Translation. Academic Press, Cambridge, Massachusetts, p123–152.

5. Choudhury D, Anand S, Naing MW, 2018, The arrival of commercial bioprinters-Towards 3D bioprinting revolution! Int J Bioprint, 4: 139. https://doi.org/10.18063/ijb.v4i2.139

6. Groll J, Burdick JA, Cho DW, et al., 2018, A definition of bioinks and their distinction from biomaterial inks. Biofabrication, 11: 013001.

7. Bonatti AF, Chiesa I, Vozzi G, et al., 2021, Open-source CAD-CAM simulator of the extrusion-based bioprinting process. Bioprinting, 24: e00172. https://doi.org/10.1016/j.bprint.2021.e00172

8. Schwab A, Levato R, D’Este M, et al., 2020, Printability and shape fidelity of bioinks in 3D bioprinting. Chem Rev, 120: 11028–11055. https://doi.org/10.1021/acs.chemrev.0c00084

9. Paxton N, Smolan W, Böck T, et al., 2017, Proposal to assess printability of bioinks for extrusion-based bioprinting and evaluation of rheological properties governing bioprintability. Biofabrication, 9: 044107. https://doi.org/10.1088/1758-5090/aa8dd8

10. Di Pietro L, Ravizza A, Vozzi G, et al., 2019, European regulatory framework for the clinical translation of bioprinted scaffolds and tissues. Biomed Sci Eng, 3: 108. https://doi.org/10.4081/bse.2019.108

11. Yu C, Jiang J, 2020, A perspective on using machine learning in 3D bioprinting. Int J Bioprint, 6: 253. https://doi.org/10.18063/ijb.v6i1.253

12. An J, Chua CK, Mironov V, 2021, Application of machine learning in 3D bioprinting: Focus on development of big data and digital twin. Int J Bioprint, 7: 342. https://doi.org/10.18063/ijb.v7i1.342

13. Freeman S, Calabro S, Williams R, et al., 2022, Bioink formulation and machine learning-empowered bioprinting optimization. Front Bioeng Biotechnol, 10: 913579. https://doi.org/10.3389/fbioe.2022.913579

14. Bone JM, Childs CM, Menon A, et al., 2020, Hierarchical machine learning for high-fidelity 3D printed biopolymers. ACS Biomater Sci Eng, 6: 7021–7031. https://doi.org/10.1021/acsbiomaterials.0c00755

15. Fu Z, Angeline V, Sun W, 2021, Evaluation of printing
parameters on 3D extrusion printing of pluronic hydrogels and machine learning guided parameter recommendation. *Int J Bioprint*, 7: 434. https://doi.org/10.18063/ijb.v7i4.434

16. Ruberu K, Senadeera M, Rana S, et al., 2021, Coupling machine learning with 3D bioprinting to fast track optimisation of extrusion printing. *Appl Mater Today*, 22: 100914. https://doi.org/10.1016/j.apmt.2020.100914

17. Jin Z, Zhang Z, Shao X, et al., 2021, Monitoring anomalies in 3D bioprinting with deep neural networks. *ACSM Biomater Sci Eng*. https://doi.org/10.1021/acsbiomaterials.0c01761

18. Chandola V, Banerjee A, Kumar V, 2009, Anomaly detection: A survey. *ACM Comput Surv (CSUR)*, 41: 1–58. https://doi.org/10.1145/1541880.1541882

19. Wang R, Nie K, Wang T, et al., 2020, Deep learning for anomaly detection. In: Proceedings of the 13th International Conference on Web Search and Data Mining. p894–896. https://doi.org/10.1145/3336191.3371876

20. Pang G, Shen C, Cao L, et al., 2021, Deep learning for anomaly detection: A review. *ACM Comput Surv (CSUR)*, 54: 1–38. https://doi.org/10.1145/3439950

21. Goodfellow I, Bengio Y, Courville A, 2016, Deep Learning. Cambridge, Massachusetts, MIT Press.

22. Chalapathy R, Chawla S, 2019, Deep learning for anomaly detection: A survey. *arXiv preprint arXiv*, 1901: 03407. https://doi.org/10.48550/arXiv.1901.03407

23. LeCun Y, Bengio Y, Hinton G, 2015, Deep learning. *Nature*, 521: 436–444.

24. Wang C, Tan XP, Tor SB, et al., 2020, Machine learning in additive manufacturing: State-of-the-art and perspectives. *Addit Manuf*, 36: 101538. https://doi.org/10.1016/j.addma.2020.101538

25. Meng L, McWilliams B, Jarosinski W, et al., 2020, Machine learning in additive manufacturing: A review. *JOM*, 72: 2363–2377. https://doi.org/10.1007/s11837-020-04155-y

26. Wu M, Phoha VV, Moon YB, et al., 2016, Detecting malicious defects in 3D printing process using machine learning and image classification. In: ASME International Mechanical Engineering Congress and Exposition. Vol. 50688. American Society of Mechanical Engineers, New York, pV014T07A004.

27. Liu C, Roberson D, Kong Z, 2017, Textural analysis-based online closed-loop quality control for additive manufacturing processes. In: IIE Annual Conference. Proceedings. Institute of Industrial and Systems Engineers, Peachtree Corners, Georgia, p1127–1132.

28. Ogunsanya M, Isichei J, Parupelli SK, et al., 2021, In-situ droplet monitoring of inkjet 3D printing process using image analysis and machine learning models. *Procedia Manuf*, 53: 427–434. https://doi.org/10.1016/j.promfg.2021.06.045

29. Gobert C, Reutzel EW, Petrich J, et al., 2018, Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging. *Addit Manuf*, 21: 517–528. https://doi.org/10.1016/j.addma.2018.04.005

30. Sing SL, Kuo CN, Shih CT, et al., 2021, Perspectives of using machine learning in laser powder bed fusion for metal additive manufacturing. *Virtual Phys Prototyp*, 16: 372–386. https://doi.org/10.1080/17452759.2021.1944229

31. Jin Z, Zhang Z, Gu GX, 2019, Autonomous in-situ correction of fused deposition modeling printers using computer vision and deep learning. *Manuf Lett*, 22, 11–15. https://doi.org/10.1016/j.mfl.2019.09.005

32. Tonnaer L, Li J, Osin V, et al., 2019, Anomaly detection for visual quality control of 3D-printed products. In: 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, Piscataway, New Jersey, p1–8.

33. Zhang J, Wang P, Gao RX, 2019, Deep learning-based tensile strength prediction in fused deposition modeling. *Comput Ind*, 107: 11–21. https://doi.org/10.1016/j.compind.2019.01.011

34. Zhang B, Liu S, Shin YC, 2019, In-process monitoring of porosity during laser additive manufacturing process. *Addit Manuf*, 28: 497–505. https://doi.org/10.1016/j.addma.2019.05.030

35. Wang T, Kwok TH, Zhou C, et al., 2018, In-situ droplet inspection and closed-loop control system using machine learning for liquid metal jet printing. *J Manuf Syst*, 47: 83–92. https://doi.org/10.1016/j.jmsy.2018.04.003

36. McKinney W, 2011, pandas: A foundational Python library for data analysis and statistics. *Python High Perform Sci Comput*, 14: 1–9.

37. Bradski G, 2000, The openCV library. *Dr Dobbs J Softw Tools*, 25: 120–123.

38. Abadi M, Agarwal A, Barham P, et al., 2016, Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv*, 1603: 04467. https://doi.org/10.48550/arXiv.1603.04467

39. Chollet F, 2018, Keras: The python deep learning library. *Astrophysics Source Code Library*, ascl-1806.

40. Pedregosa F, Varoquaux G, Gramfort A, et al., 2011, Scikit-
learn: Machine learning in Python. *J Mach Learn Res*, 12: 2825–2830.

41. Bonatti AF, Fortunato GM, Lapomarda A, *et al.*, 2021, Advanced Firmware and Hardware for Multiscale and Multimaterial Bioprinting. Proceedings of the Seventh National Congress of Bioengineering.

42. Zhao M, Bu J, Chen C, 2002, Robust background subtraction in HSV color space. In: Multimedia systems and Applications, Vol. 4861. International Society for Optics and Photonics, United States, p325–332. https://doi.org/10.1117/12.456333

43. Simonyan K, Zisserman A, 2014, Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv*, 1409: 1556.

44. He K, Zhang X, Ren S, *et al.*, 2016, Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. p770–778.

45. He K, Zhang X, Ren S, *et al.*, 2015, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision. p1026–1034.

46. Kingma DP, Ba J, 2014, Adam: A method for stochastic optimization. *arXiv preprint arXiv*, 1412: 6980. https://doi.org/10.48550/arXiv.1412.6980

47. Selvaraju RR, Das A, Vedantam R, *et al.*, 2016, Grad-CAM: Why did you say that? *arXiv preprint arXiv*, 1611: 07450. https://doi.org/10.48550/arXiv.1611.07450

**Publisher’s Note**

Whioce Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.