Injection Molding

Dec 03, 2021

OBJECTIVES

Mass production of injection molded parts has become incredibly prevalent in manufacturing. While injection molding machines have become more effective through technological innovations, it is undeniable that machines can eventually fail, or have issues which cause deterioration in the quality of production. Checking the quality of each part by eye and hand can be labour intensive and time-consuming, especially in industrial applications where thousands of parts are produced every day.

In order to assess the quality of the parts from injection molding machines based on a variety of factors, we have developed machine learning algorithms To assess the following characteristics of each part:

1. Identify when parts have flashes or voids, including completeness of circular section
2. Predict the tensile strength and maximum deflection of beam
3. Identify the quality of transparency of the circular section

Through data collection and creating machine learning algorithms, we have been able to create models which are able to predict and identify these key features which indicate the quality of a part.

Ultimately, the hope is that the algorithms would be used by attaching a camera to inspect the parts which would be produced by the machine. Using the algorithm on that camera would allow the user to determine whether or not the machine was functioning as intended, and to see if there was any need for human intervention. This can help save on costs by not requiring someone to manually sort or monitor the machine under large loads.

EXPERIMENTAL METHODS

To collect the data necessary to train the machine learning model, we designed a process that would allow us to collect diverse data. In order to do that, first, our team printed out 200 parts. We aimed to produce three types of parts: perfect parts (with no defects), which were produced at the optimal temperature and pressure (240 degrees C and 56MPa), parts with voids in the circle or other places, and parts with flash, which were produced in high pressure and temperature. We produced 120 parts using the optimal temperature and pressure, 20 parts using different pressures, which varied between 45-60MPa and constant temperature at 240 degrees C, 20 parts using 2 different temperatures 230 and 260 degrees C and constant pressure at 55MPa and 40 parts in which both temperature and pressure varied. These parts were printed in the Manufacturing Laboratory by the BOY Injection printer in FF219.

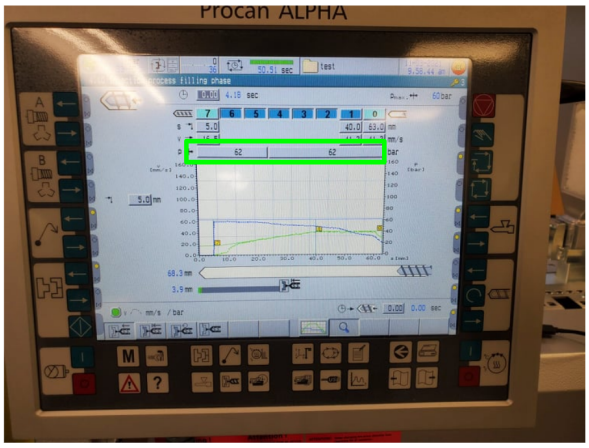
Afterwards, we collected more data on the parts by taking pictures, performing tensile tests and calculating the beam deflection.

Injection Molder Data Collection

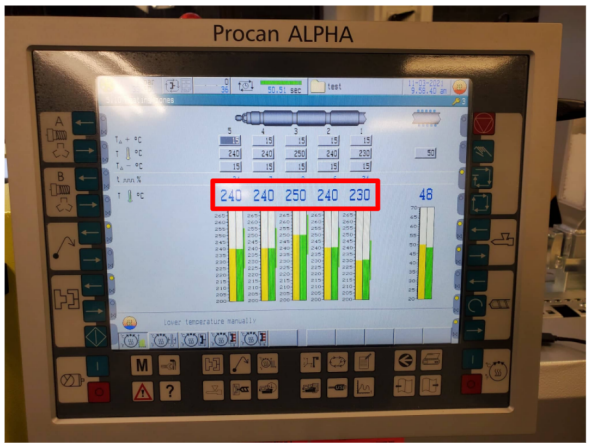
Critical data from the injection molder was collected, namely back pressure, front pressure, and temperature. However, the temperature and pressure inputted into the machine was not

necessarily the real pressure and temperature. This occurs mainly because of two factors: it takes time for the machine to heat up and cool down, and a long time of operation can cause variance in the temperature and pressure inside the machine. Unfortunately, the machine was unable to output the real time data while it was operating, so our team was not able to get the real pressure and temperature. However, we were able to estimate some of these variances by inspecting the part. For example, if the machine read a constant pressure, but we were noticing that the part was slowly starting to lose its form, we could determine that the pressure within the machine was actually falling. At this point, we would change the input value to the machine, to forcibly bring the pressure back up.

The data was read from the green and red box shown in the figure below:



Pressure



Temperature

The pressure collected from the machine was in Megapascals, and the temperature was in degrees Celsius. For the temperature, we considered an error of ± 1 degree Celsius, and the pressure variance was considered to be ± 1MPa. The raw data of the temperature and pressure for each part can be found in Appendix A.

The following table provides a breakdown of the 200 parts printed:

Table 1: Part Breakdown

Feed Pressure	Temperature	Prediction Scenario			# Product
		Void	Flash	Good	
Low	Low	✓			10
Regular	Low				10
High	Low		✓		10
Low	Regular	✓			10
Regular	Regular			✓	120
High	Regular		✓		10
Low	High	✓			10
Regular	High				10
High	High		✓		10

Once the parts were printed, they were labeled based on the machine parameters used to produce them, as well as what defect they had. The labels were color-based. For example, a red label would mean that the pressure used to print the part was significantly different from the optimal pressure. The same was done for temperature, and when both temperature and pressure were varied. The table below outlines the label type and the parameters which were changed from the optimal values:

Table 2: Labels and Parameters

Label	Parameters Changed

Red Label	Pressure varied
Green Label	Temperature varied
Red/Green Label	Both pressure and temperature varied
No Label	Suggested pressure and temperature

The following sections outline the methods used to collect further data from the part in order to achieve the goals for our project.

Goal 1: Identification of void, flash and circle completeness

Each object had photos taken by the IPEVO VZ-R camera in the FF lab up against a black background. The black background was used to bring out the edges of the part as much as possible. The objects were rotated in different orientations to improve robustness of the algorithm, and the images were taken with consistent lighting as well. A few examples can be seen below.

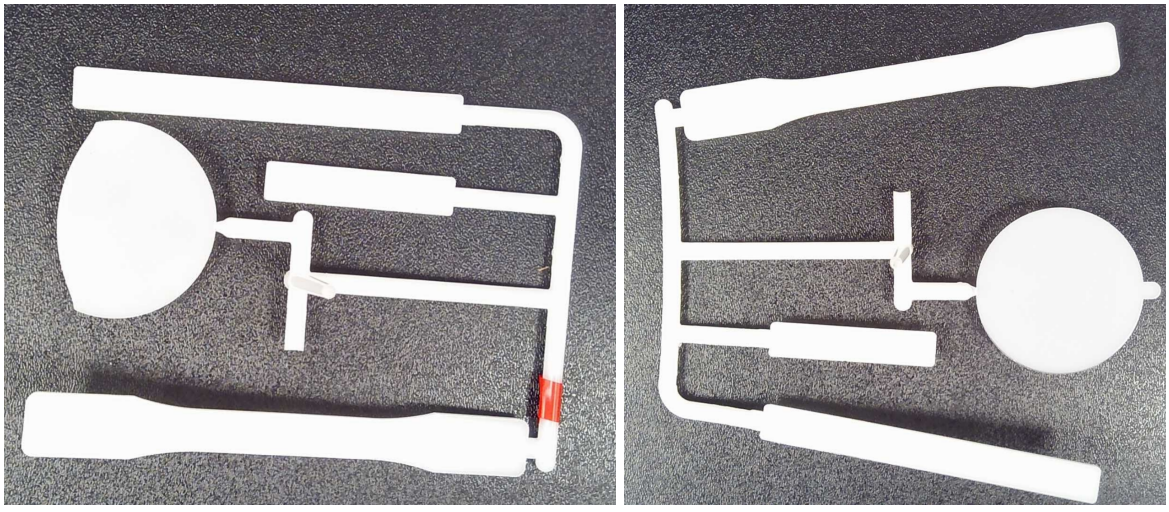


Figure 1: Example Images of Part

These images were taken and labeled in different folders based on labelled (Bad Parts) or non-labelled (Good Parts), so that they could be easily distinguished to create and design a CNN algorithm.

Goal 2: Identify Tensile Strength and Maximum Deflection of Beam

Tensile Tester Data Collection

A specific portion of the part was designed to be tested for tensile strength, the part shown in the green in the picture below.

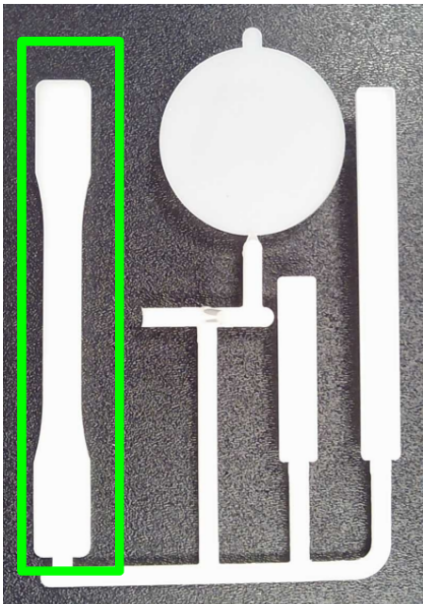


Figure 2: Part for Tensile Testing

To perform the tensile test we followed the following process:

1. Trim off the test part, shown within green box above
2. Measure and record the thickness and width around middle neck of the part
3. Clamp the piece in the tensile machine, making sure it is aligned properly

4. Input the test part ID, the thickness, width of the part, as well as the label (Green/Red/Good) into the machine for later organized result outputs
5. Attach strain-clamp on the neck of the part and balance the machine reading
6. Press the start button. Once the machine has its first sudden stop, remove the strain-clamp, and press start to begin the tensile test initiation
7. At point when the piece fails under large stress, remove the piece and return the machine to its home position

Some of the tensile tests failed because the piece did not break after the tensile testing machine had reached its setted maximum length of actuation. When this occurred, the test was discarded and done again with adjusted maximum length of actuation, for another part made under the same conditions.

Material properties of the piece, such as yield strength, ultimate strength, and percentage elongation, for Red, Green, Red/Green, and Good parts were collected and outputted from the tensile tester. This data can be seen in Appendix A, and more information on this data can be found in Appendix C.

Maximum Deflection

To identify the maximum deflection of the part, again, only a portion of the part is used. It can be seen in the picture below:

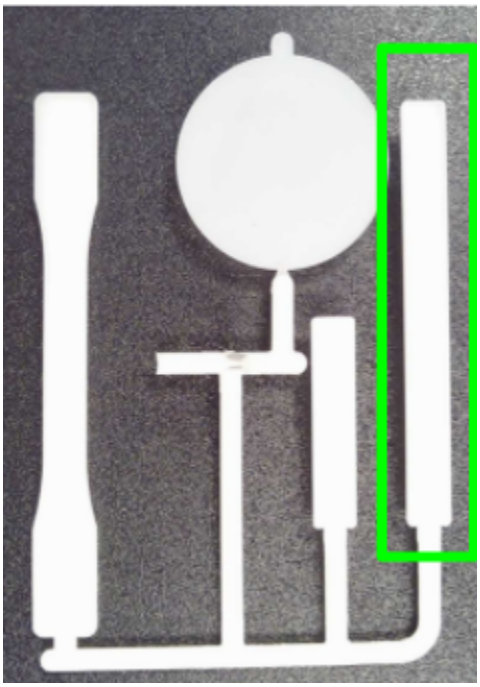


Figure 3: Deflection Part

A setup was put together including clamping the part between two slots on shelves, placing a meter stick positioned perpendicular relative to ground to measure deflection, and clipping a weight to the free end of the beam. The weight used was 185 grams. The setup can be seen below:

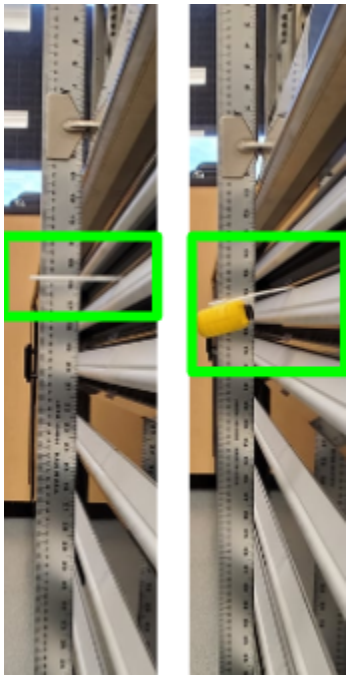


Figure 4: Beam Deflection Setup

On the left, the beam without any deflection is seen. The current un-bended position was marked as the natural resting state.

On the right, the beam with the weight is added, the deflection of the end point of the beam is measured and recorded by taking the difference between the bent side and the natural resting state. Because each part was clamped in the same location, the 0 position was consistent throughout all of the testing.

Again, only a portion of the dataset was used, and this sample was used to represent the other parts which were printed under the same machine conditions. The data for the tensile testing can be found in Appendix A.

Goal 3: Identify Transparency of the Circular Section

To perform this portion of the data collection, the following setup was used:

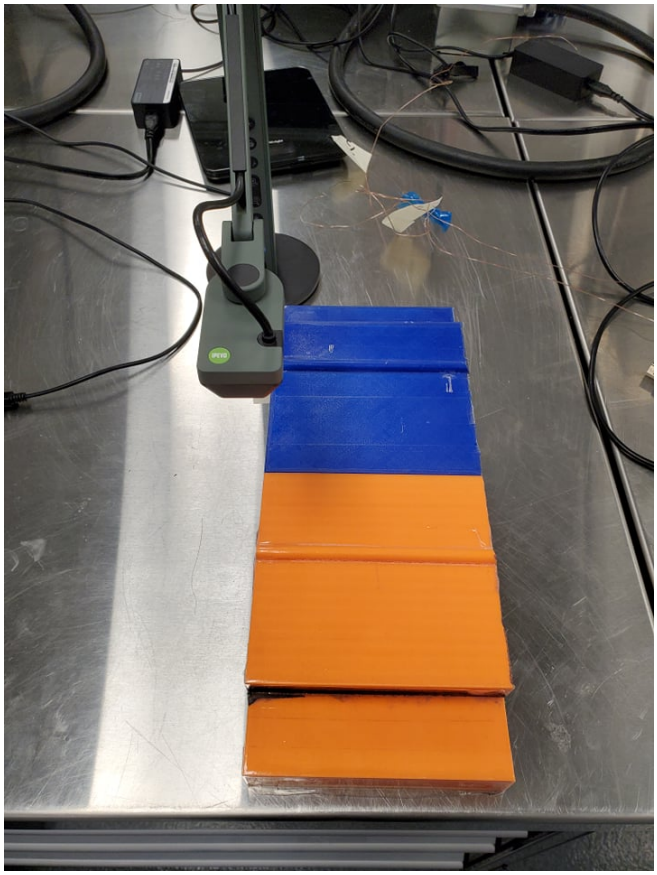
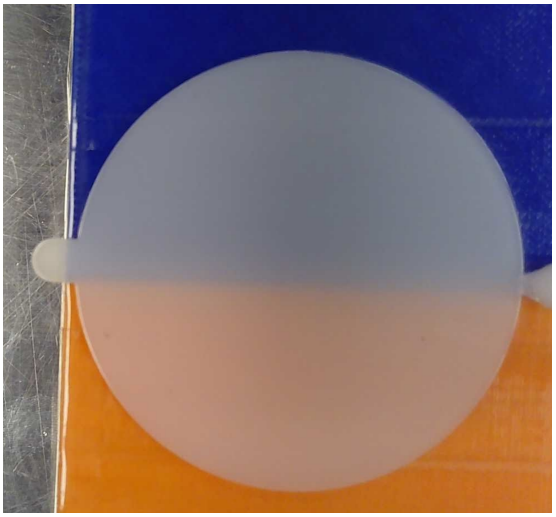


Figure 5: Transparency Setup

The circular part was placed on the intersection of the orange and blue blocks. The photo is taken again using the webcam. Sample images taken are as follows:



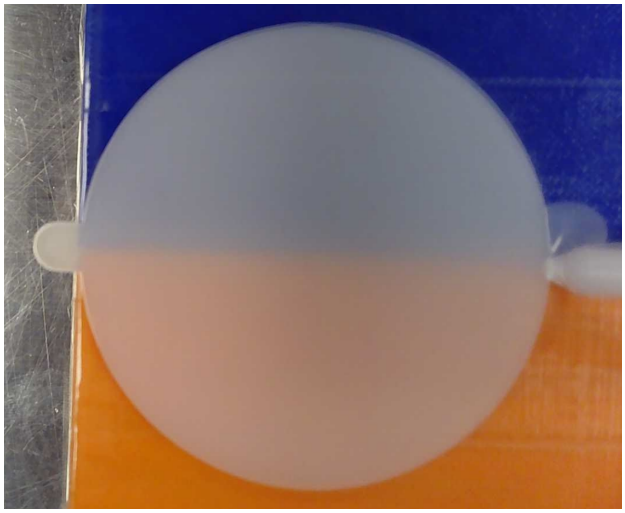


FIgure 6: Sample Transparency Photos

These images were verified to be bad or good transparency based on the intensity of the color seen through the circle. From the images above, the right is seen to have bad transparency as the colors are more muted, and the left has good transparency. These again were taken based on a sample size which represents the data that we have received. However, these images were not very effective when being used in the dataset due to the resolution of the camera. The data for whether or not these were good or bad transparency can be found in Appendix A.

Data Analysis

Before any coding or algorithm building could be done, the data collected had to be processed. Given that only a few batches of each type of part were tested, there was a lot of data to parse and manipulate before we were able to properly create algorithms for them.

Refine Data:

Injection Molder Data

The raw data can be found in appendix A. Because of the discontinuous lab schedules and insufficient part management, some data for several tested parts were missing. Moreover, there was no way to retrieve real-time readings from the injection molding machine itself, as any attempt to connect a drive to the machine did not yield data which could be used for the project.

Tensile Testing and Beam Deflection Data

As stated in the Experimental Methods section: Goal 2, as well due to the time constraints, only a batch of each part was used for actual testing. Given that leaves a portion of the data to be lacking because they were not tested, these missing values would have to be dealt with. However, the tensile testing, as mentioned before, had some failed tests which caused us to lose data on some parts whenthey did not fail as expected. Because these parts were also tested for beam bending, these values would result in missing values in our dataset.

A quick look at some of the features in each dataset, where Tensile Strength is in MPa and Max Deflection is in mm:

Table 3: Tensile and Deflection Overview

	Average		Min		Max	
	Tensile Strength	Max Deflection	Tensile Strength	Max Deflection	Tensile Strength	Max Deflection
Red	24.83434783	2.563636364	24.09	1.6	26.27	4
Green	24.485	2.628571429	23.96	2	25.08	3
Good	24.53921053	1.989189189	23.62	1.2	25.55	2.8

The values in the table above were before any post processing occurred.

An example of the missing data can be seen in the table below:

Table 4: Sample Missing Data

ID	Pressure(front)	Pressure(back)	Temp	Label	thick	width	Tensile Strength (MPa)	Deflection
1		56	240		2.993	12.6	24.22	
2			240		2.965	12.61		
3			240		3.002	12.61	23.69	41.8
4			240		2.993	12.68	23.76	42
5			240		2.963	12.67	24.42	42.2
6			240		2.954	12.57	25.55	42.6
7			240		2.958	12.66	24.76	42
8			240		2.961	12.54	24.58	42
9			240					
10			240					42.8
11			240					
12			240					
13			240					
14			240		2.994	12.59	24.7	

As can be seen, the pressure columns, thickness, width, tensile strength, and deflection all have missing values. An example of the failed tensile test can be seen for item #10 and #19.

To deal with this data loss, a simple imputer function from the SciKit Learn Python Library was used. Essentially, the mean of all the existing values within the column was taken, that mean value was then used to fill in the rest of the missing data. The imputed data was used to increase the size of the viable dataset for testing and algorithm building purposes.

Post-Imputed Data

A sample of the data after being imputed is shown below:

Table 5: Sample Imputed Data

	Pressure(front)	Pressure(back)	Temp	thick	width	Tensile Strength (MPa)	Deflection
0	54.0	56.000000	240.0	2.993000	12.600000	24.220000	42.248485
1	54.0	48.862745	240.0	2.965000	12.610000	24.635672	42.248485
2	54.0	48.862745	240.0	3.002000	12.610000	23.690000	41.800000
3	54.0	48.862745	240.0	2.993000	12.680000	23.760000	42.000000
4	54.0	48.862745	240.0	2.963000	12.670000	24.420000	42.200000
...	...	...	...	...	...	...	...
169	65.0	56.000000	240.0	2.970918	12.771918	24.635672	42.248485
170	65.0	56.000000	240.0	2.970918	12.771918	24.635672	42.248485
171	42.0	40.000000	240.0	2.970918	12.771918	24.635672	42.248485
172	48.0	49.000000	240.0	2.970918	12.771918	24.635672	42.248485
173	48.0	40.000000	240.0	2.970918	12.771918	24.635672	42.248485

After the data imputation was complete, a quick overview of the various data we got can be seen in the histogram below.

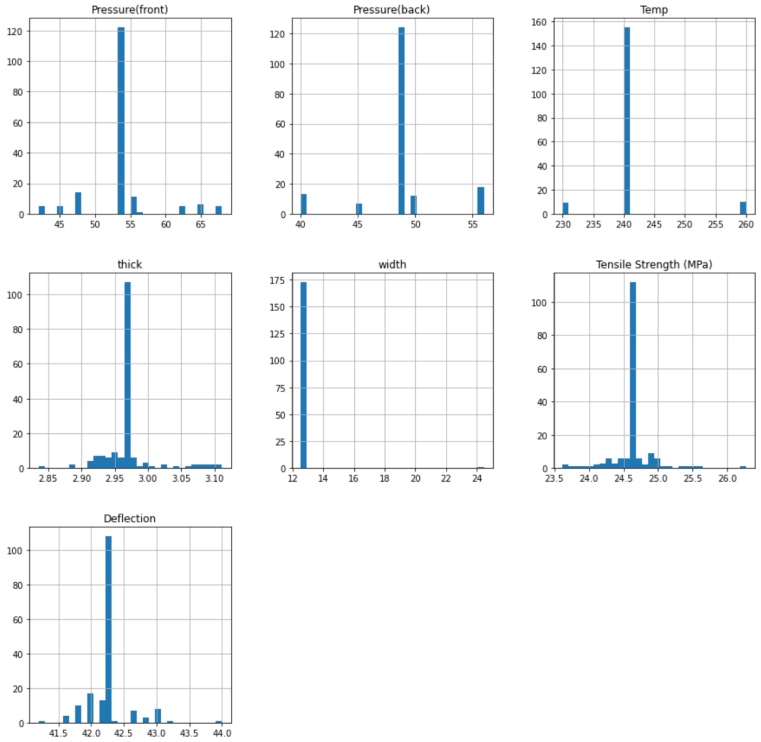


Figure 7: Histogram of Imputed Data

It can be seen that the range of data collected isn't the most diverse, which is to be expected given the nature of the project and the data we are collecting.

Feature Scaling

Because the data spread is quite large, as temperature values are high at 240 ℃, and thickness values are quite small, in the single digit values, feature scaling using the StandardScaler from the SciKit Learn library was used to scale down the data for a more robust prediction model.

Correlation Analysis

Before creating algorithms or models, a correlation analysis was performed on the features which were assigned to the parts. Unfortunately from the correlation matrix, the features which we aim to predict, and the features we aim to use as inputs do not have the strongest correlations between each other.

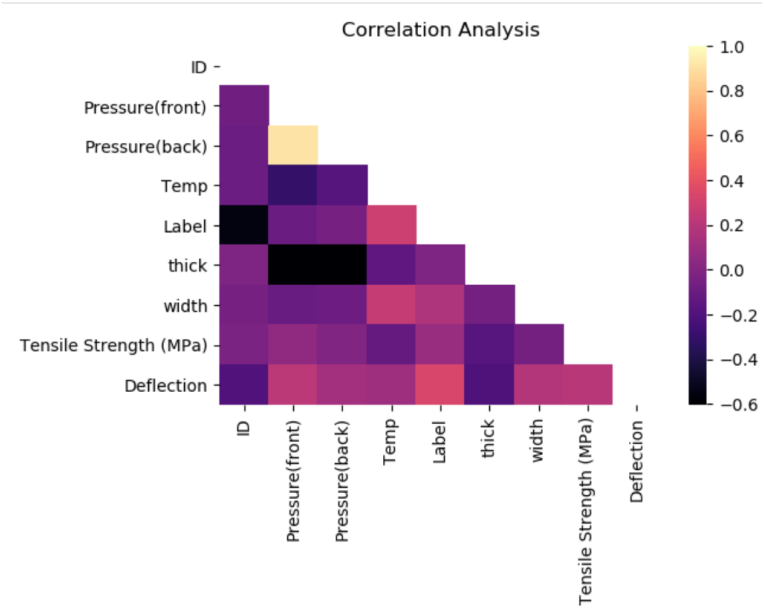


FIgure 8: Correlation Matrix

Taking a closer look at the correlation between the features: We can generate the following datasets through the Pandas Python library

Tensile Strength:

```
Tensile Strength (MPa)    1.000000
Deflection                0.200720
Label                     0.082306
Pressure(front)           0.050663
Pressure(back)           -0.013435
ID                       -0.031712
width                    -0.062640
Temp                     -0.120207
thick                    -0.173836
Name: Tensile Strength (MPa), dtype: float64
```

Figure 9: Tensile Strength Correlations

As seen from the Tensile Strength, the strongest correlations are between the front and back pressures, as deflection and label are not input features. Unfortunately, width, temperature, and thickness have negative correlations to the tensile strength. However, we will see later on that these correlations do not necessarily spell doom for creating machine learning models.

Deflection:

```
Deflection                1.000000
Label                     0.341548
Pressure(front)           0.221954
Tensile Strength (MPa)    0.200720
width                     0.190768
Pressure(back)            0.123119
Temp                      0.109745
ID                       -0.197515
thick                    -0.204527
Name: Deflection, dtype: float64
```

Figure 10: Deflection Correlations

As seen from the deflection, the strongest correlations are with the front pressure, tensile strength, and width. Moreover, the back pressure and temperature are also decently correlated to the deflection. This result gives some confidence in our model being able to predict the deflection relatively well from the input parameters which we plan to use.

Image Data

As mentioned in the Experimental Methods: Goal 1, the image used was taken against a uniform background to best bring out the edges of the part. Moreover, the shape was distorted and rotated to increase the robustness of the dataset. Furthermore, shear, flip, and zoom after processing was applied to further develop the size of the dataset to help the image recognition improve. However, there were some issues due to reflection from ceiling lighting and off the glossy surface of the background. Light reflection covers a large portion of the image and this becomes an issue for our later model training that limits the model accuracy below 64% from a one-day iteration. However, a resolution was found which will be shown in the “CNN shapes” section later.

Identification of Key Features

Due to the nature of the project, there were only a few features which would be relevant if our application for the process was to be applied. On this end, the input parameters: front pressure, back pressure, and temperature were the key features for determining the tensile strength and the maximum deflection of the beam in our first attempt. To seek potential improvement, the thickness and width were also incorporated to try and add further features to the predictive method for iteration, resulting in better results from metrics of MSE.

By first using these features, several machine learning algorithms were created.

Algorithms and Testing

Tensile Test and Deflection Predictor

After the data was collected and imputed, two predictors were to be built to predict tensile strength and deflection based on various input parameters, which include temperature, pressure, back pressure, thickness, and width. The data after the preprocesses was standard scaled and split into 80% train, and 20% test sets.

To find out which method of prediction would be the best to use in our final implementation, various different algorithms were attempted to determine which one would be the best performing. In this section, each attempted algorithm will be discussed and the final results of each will be shared.

Regression ANN

An ANN regressor was used to predict tensile strength, and a separate regression was used to predict deflection/beam bending of the part. Both ANN Regressors used 64 units, had an input dimension of 5, with ReLU activation function embedded in each layer. The layers were varied and tuned to maximize performance, so some of the layer parameters were changed based on the data the regressor was predicting.

Tensile Strength Regressor

The exact architecture of the tensile regressor is shown below. For the ANN algorithms, the MSE was used to measure the performance of each algorithm. Detailed ANN architecture can be viewed under Appendix D.

After running the algorithm, the following error and performance was found:

```
The average error for tensile ANN regressor is  -0.12346455783478023
2/2 [==============================] - 0s 3ms/step - loss: 2.0705 - mean_squared_error: 2.0705
[2.07047963142395, 2.07047963142395]
```

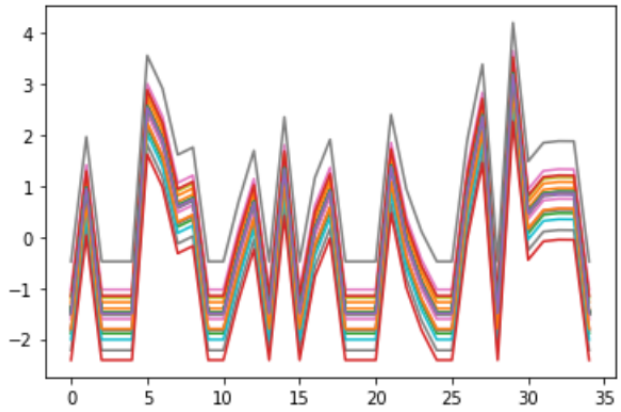


Figure 11: Error of Tensile ANN

It can be seen the average error is -0.123, with an MSE of 2.0705. Here, we can conclude that the algorithm performs generally well, but there could be some improvements made including varying number of hidden layers, as well the number of neural nodes in each hidden layer .

The exact architecture of the tensile regressor is shown in Appendix D.

Maximum Deflection Regressor

A few notable differences are the units in the second layer, being 12 here for the deflection instead of 8 for the tensile test, and using 300 epochs instead of 100. Again, these values were tuned through iteration to bring out the best performance for these models.

After running the algorithm, the following error and performance was found:

```
The average error for tensile ANN regressor is  -17.73221625389991
2/2 [==============================] - 0s 4ms/step - loss: 0.5043 - mse: 0.5043
[0.5042535066604614, 0.5042535066604614]
```

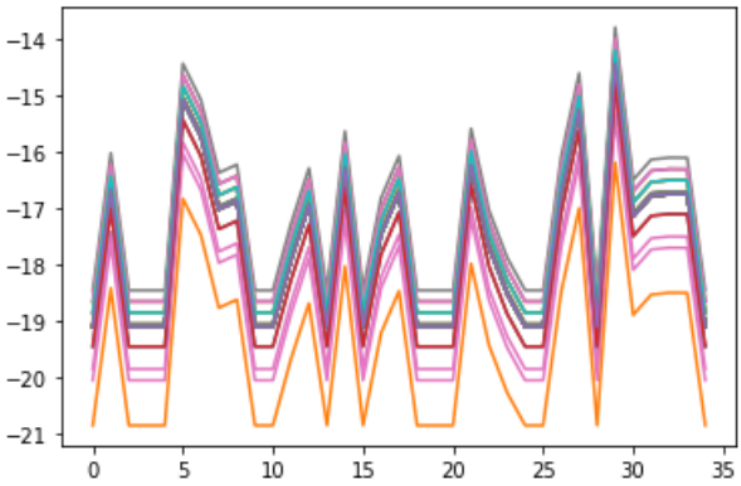


Figure 12: Error of Deflection ANN

It can be seen the average error is -17.732, the loss is 0.5043, and the Mean Squared Error is 0.5043. Here, we can conclude that the algorithm performs quite well, with little loss and a relatively small MSE score compared to the ANN for tensile testing.

While the correlation matrix from the data analysis showed that the relationship wasn't necessarily the strongest for the features which the model was based on, the models aren't actually the worst performers. Our creation and testing of our model shows that despite a low correlation, a decent prediction algorithm can be created and used.

For a further check, we ran a prediction test for each ANN algorithm, to verify the output would lie within reasonable ranges. With this, we would compare the predicted value with an actual value

from the dataset to see if the algorithm is capable of predicting something reasonable.

```
# Prediction for Tensile Strength
NewXt=[[55,54,240,3,12.5]]
ScaledNewXt=sc.fit_transform(NewXt)
Prediction=InjMachineT.predict(ScaledNewXt)
print('The prediction for tensile ANN regressor is ',Prediction)
```

```
The prediction for tensile ANN regressor is  [[21.327988]]
```

```
# Prediction for Deflection
NewXd=[[55,54,240,3,12.5]]
ScaledNewXd=sc.fit_transform(NewXd)
Prediction=InjMachineD.predict(ScaledNewXd)
print('The prediction for deflection ANN regressor is ',Prediction-40)
```

```
The prediction for deflection ANN regressor is  [[1.0859375]]
```

Figure 13: ANN Regressor Predictions

We created two new input data, NewXt and NewXd for the tensile and deflection respectively. We can see the tensile strength was predicted to be 21 MPa, and the deflection was 1.085 mm. Looking in the data we collected experimentally, we find the datapoint for NewXt and NewXd actually had a tensile strength of 23.69 MPa, and a deflection of 2.08 mm, a little bit of deviation. After performing more of these tests by predicting our values, we find that generally the algorithm does well.

We will continue to look for algorithms which may be able to perform better, especially focusing on creating a better model for tensile testing.

Random Forest

A random forest was used as another method for creating an algorithm for predicting tensile strength and beam deflection. Here, the main parameter which was changed as the number of

predictors. 30 was chosen after tuning the model and attempting to minimize its R2 score. For comparison against other algorithms, its MSE score was also found. The tensile random forest performed yielding a score of 0.084 and the deflection yielded a score of 0.174. The following shows a general overview of the architecture behind the algorithm.

```
rf=RandomForestRegressor(n_estimators=30)
rf.fit(Xt_train,Yt_train)
Yt_pred = rf.predict(Xt_test)
```

```
rf=RandomForestRegressor(n_estimators=30)
rf.fit(Xd_train,Yd_train)
Yd_pred = rf.predict(Xd_test)
```

Figure 14: Forest Model Overview

Linear Regression

A simple Linear regression was built using the SKLearn library. The Linear regression actually performed very well for the Tensile strength, yielding an MSE score of 0.088. For the deflection, linear regression yielded an MSE score of 0.146. Nothing was done out of the ordinary for this rudimentary linear regression. The following shows a general overview of the architecture behind the algorithm.

```
regressor = LinearRegression()
regressor.fit(Xt_train, Yt_train)
Yt_pred = regressor.predict(Xt_test)
```

```
regressor = LinearRegression()
regressor.fit(Xd_train, Yd_train)
Yd_pred = regressor.predict(Xd_test)
```

Figure 15: Linear Regression Model Overview

SVR

SVR was also used as an algorithm to create a model for prediction. Here, the library was imported from SK Learn, using the rbf kernel. The following shows a quick overview of the architecture:

```
Tregressor = SVR(kernel = 'rbf')
Tregressor.fit(Xt_train, Yt_train)
```

```
Dregressor = SVR(kernel = 'rbf')
Dregressor.fit(Xd_train, Yd_train)
```

Figure 16: SVR Regression Model Overview

For further robustness on this model, the same prediction method which was used on ANN was used. As a recap, a datapoint NewXt and NewXd was put into the SVR model to compare its output to experimentally collected data.

```
#Tensile Test Prediction
NewXt=[[55,54,240,3,12.5]]
ScaledNewXt=sc.fit_transform(NewXt)
Result=Tregressor.predict(ScaledNewXt)
print("The Tensile Strength will be",Result, "MPa")
Result=Tregressor.predict(Xt_test)
avgError=np.average(Result-Yt_test)
print(avgError)
Score=mean_squared_error(Result,Yt_test)
print(Score)

model_score_ten['SVR']=Score
```

```
The Tensile Strength will be [24.52903724] MPa
```

```
#Beam Bending
NewXd=[[55,54,240,3,12.5]]
ScaledNewXd=sc.fit_transform(NewXd)
Result=Dregressor.predict(ScaledNewXd)
print("The deflection will be",Result-40, "mm")
Result=Dregressor.predict(Xd_test)
avgError=np.average(Result-Yd_test)
print(avgError)
Score=mean_squared_error(Result,Yd_test)
print(Score)

model_score_def['SVR']=Score
```

```
The deflection will be [2.1397742] mm
```

Figure 17: SVR Regression Predictors

Recall that the actual datapoint at this point actually had a tensile strength of 23.69 MPa and a deflection of 2.08 mm. While the tensile strength predictor worked relatively well, the beam bending prediction had some error. Overall, the tensile algorithm had a MSE of 0.102, and the deflection algorithm had a MSE of 0.142.

Regression Recap

Ultimately, we had quite a few algorithms which performed well, some more than others. We can conclude that sometimes metrics such as error and correlation doesn't necessarily mean that the model itself will be unusable. As we have proved a few times, the models we created were actually quite serviceable despite having low correlations and high errors on paper.

Below is a overview of the various algorithms created:

Tensile Strength

	MSE Score
Random Forest	0.084285
Linear Regression	0.088358
SVR	0.102039
ANN	2.070480

Deflection

	MSE Score
Random Forest	0.157274
Linear Regression	0.146469
SVR	0.141732
ANN	0.504254

Figure 18: Model Overviews

We can recommend that the Tensile predictor use the random forest, while the deflection uses the SVR. However, we can also say with confidence that the ANN is a reasonable method to use, and by extension, all other algorithms are also feasible.

Defect Classifiers

Classifier Data Pre-processing

As a second approach to the problem, we have also built six ML classifiers and one DL classifier. Starting with data preprocessing, we separated the entire dataset into three sub-datasets as

Green (Varied Temperature), Red (Varied Pressure), and Good (Recommended Temperature and Pressure). Then boxplots were applied to eliminate outliers for each sub-set as shown below:

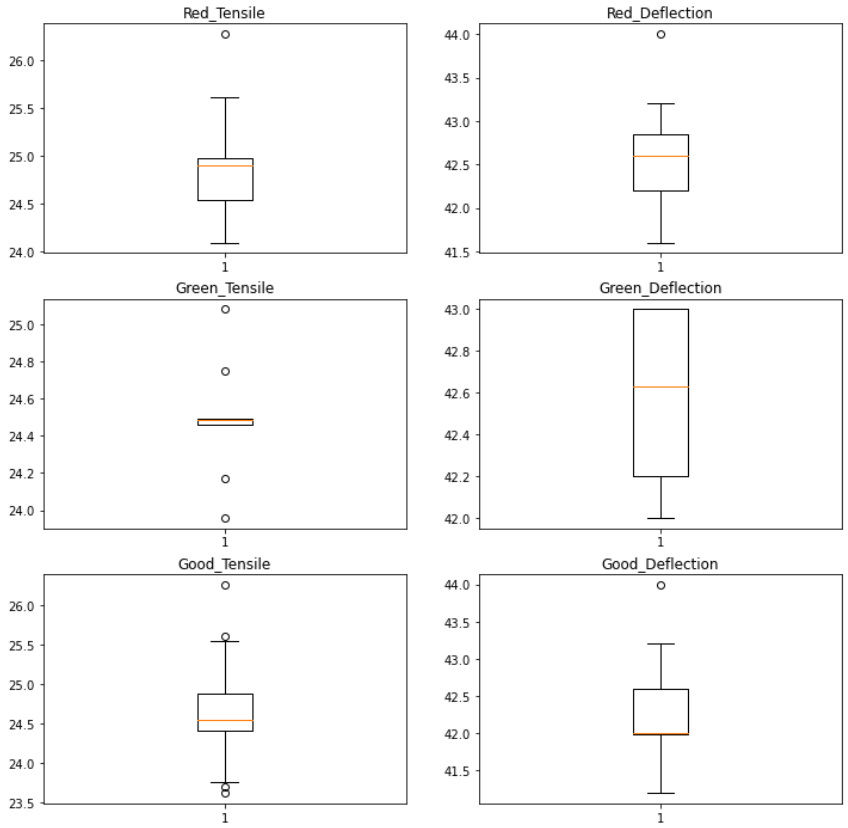


Figure 19: Box Plots

Next, red and green were encoded to be 0 for bad (red) and 1 for good(1), and imputed missing data for each sub-dataset. Lastly three clean sub-datasets were concatenated to a complete clean dataset for further standard scaling and fed into model training.

Classifiers Construction and Optimization

For refining model performance, iteration of testing is necessary. For models that have multiple tuneable hyper-parameters, we used GridSearchCV, to improve the model accuracy by

optimizing the selection of hyper-parameters. For models that have less hyper-parameters, simple Pipeline Cross Validation was used to evaluate their performance.

For ANN, optimization of the neural network was implemented, and finalized the ANN architecture using three hidden layers and 150 epochs. Detailed ANN architecture can be viewed in Appendix D.

All classifiers with the corresponding accuracy by cross-validation is shown below:

	Model Accuracy
Naive Bayes	0.60
Random Forest (GSCV)	0.52
LDA	0.64
KNN	0.64
SVC_rbf	0.64
Logistic Regression	0.64
ANN	0.64

Figure 20: Classifier Overview

From the table, 64% is the bottleneck and this is most likely due to the quality of the data as well as the parameter inputs.

Convolutional Neural Networks

Part Shape Detector

For the first CNN, it was designed to distinguish whether a part is good or bad, where a bad part can have any flash or void present. Sample photos are shown below.

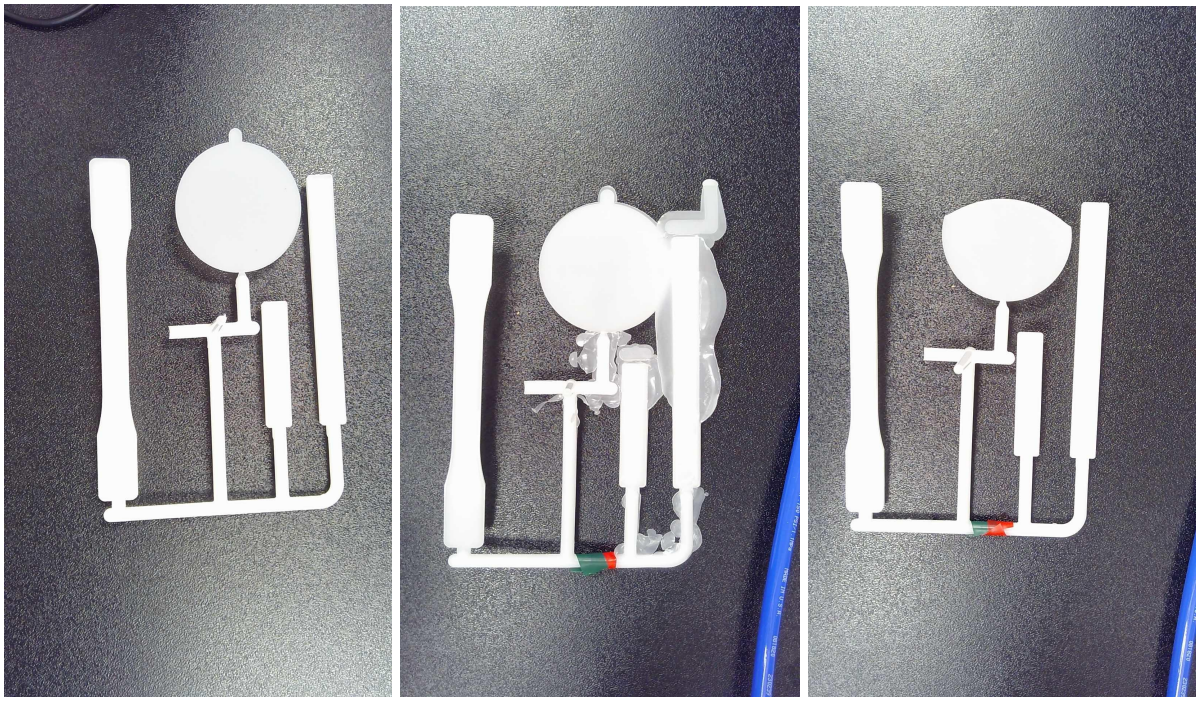


Figure 21: Images of Parts: (Left: Good, Center:Flash, RIght:Void)

With our first approach, the main structure of the CNN is shown below:

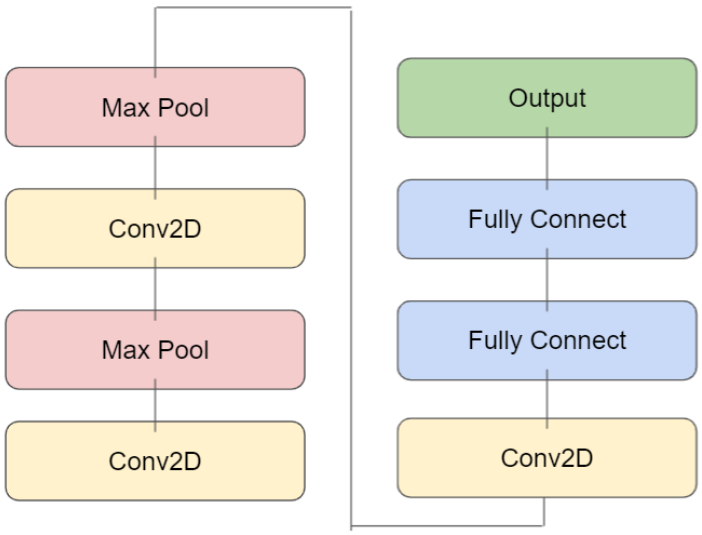


Figure 22: Structure of Image CNN

Over several time-consuming iterations with different combinations of hidden layers and nodes within each layer, and activation functions, the highest accuracy model was unable to exceed 60%.

However, we later considered using computer vision to understand the process instead of thinking about it from a human's perspective to improve the model. Because from our human eyes, we can easily distinguish the difference between whiteness of part and light reflection, however, this might be difficult for computers. Recall that Max Pooling will keep the largest value, and drop all other values within the size of the map. In this case, the white lighting reflection will be kept, and black background is dropped, which results in amplifying the source of error, causing greeted confusion.

After we realized the root of this issue, we then modified our architecture, using a combination of Average Pooling and Max Pooling shown in the following figure:

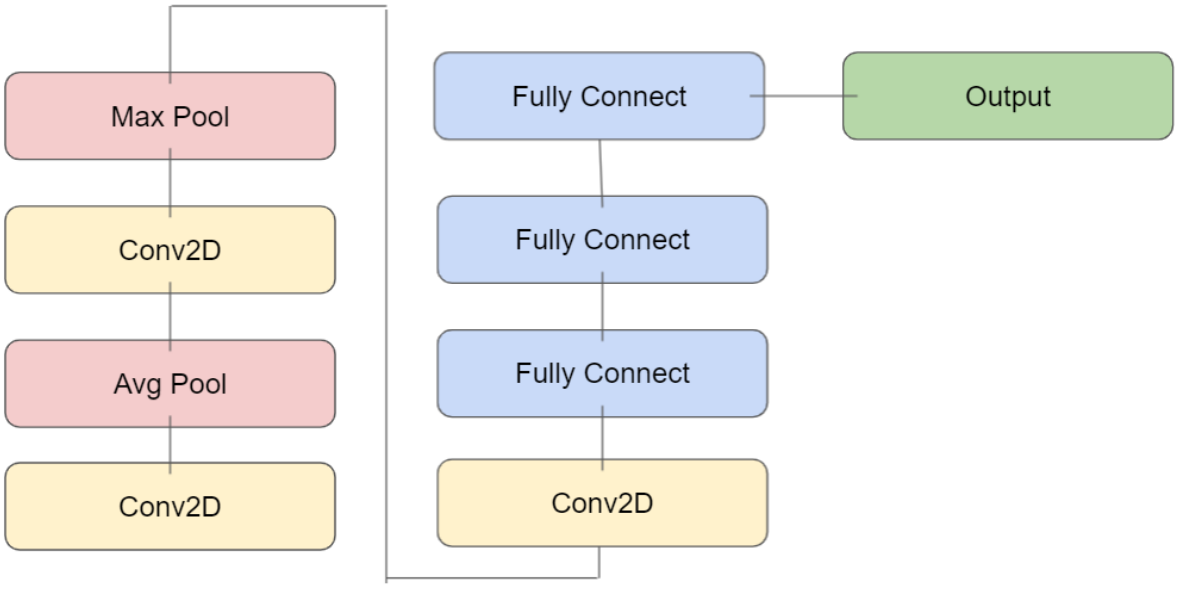


Figure 23: Structure of Improved Image CNN

After several iterations, the improved CNN architecture was built, resulting in an accuracy of 92.5%. Detailed CNN architecture can be viewed under Appendix D.

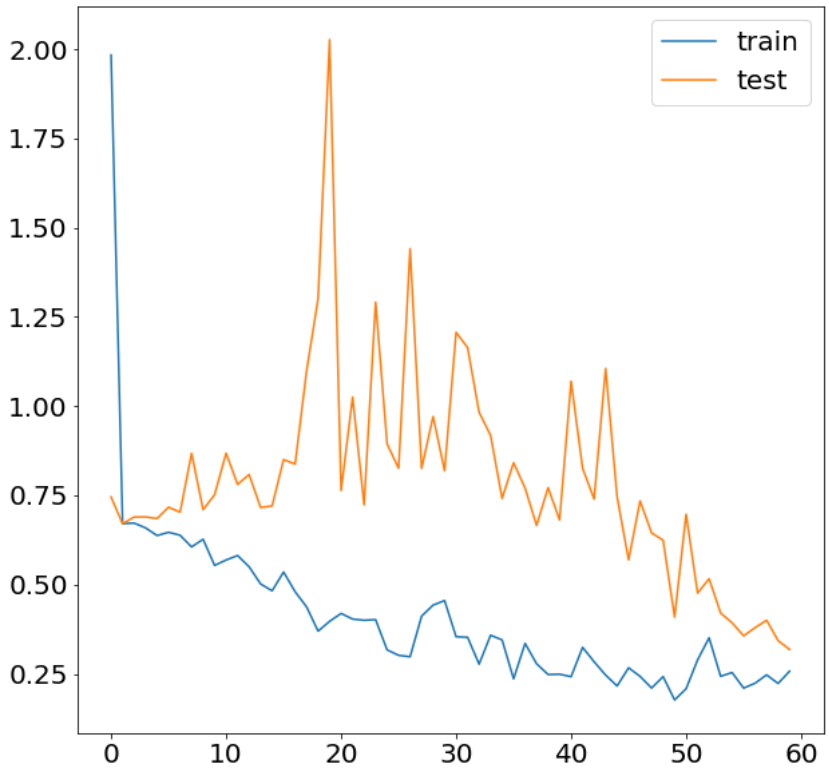


Figure 24: Loss vs Epochs, Graph of Train and Test

To seek deeper insight of the model performance, a loss vs. epochs diagram was plotted. Loss was dramatically increased at 20 epochs, and settled down after 50 epochs

Circular Part's Transparency Inspector

Based on the workflow of our monitoring system, after eliminating void/flash parts, a CNN that determines the circular part's transparency was built. Similar to the construction for the previous CNN, iterations for architecture were needed to improve model accuracy. The main flow for this CNN is shown below:

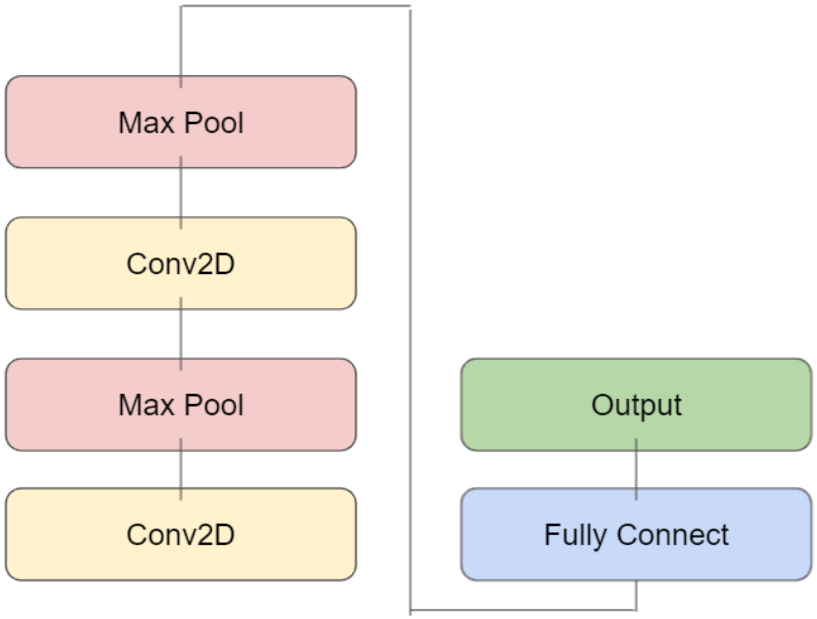


Figure 25: Structure of Transparency CNN

The accuracy of this CNN was 75%; however, with the limited amount of input image data, this accuracy is considered less reliable, with possible overfitting. The detailed CNN architecture can be viewed under Appendix D.

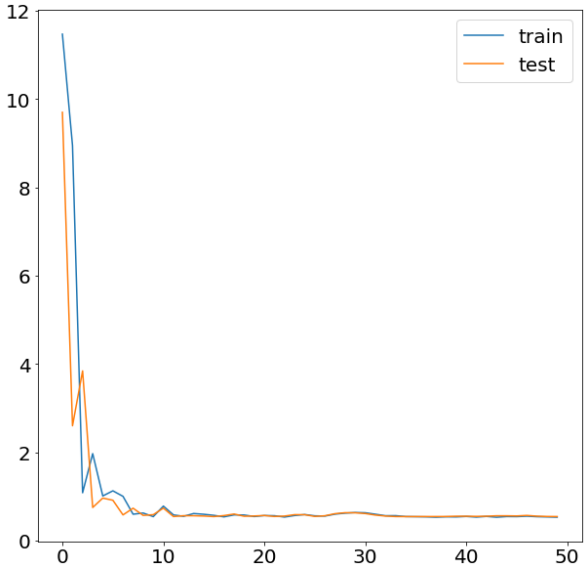


Figure 26: Loss vs Epochs, Graph of Train and Test

From the chart above, loss dramatically decreased before reaching 10 epochs, and stabilized afterward. For future development, epochs of 15 can be assigned instead of 50 to release computational resources.

Conclusion

Based on our findings and the accuracy of the machine learning algorithms, our team proposes the following recommendations:

- To predict the tensile strength of the part produced, the random forest algorithm should be used, since it has a MSE score of 0.084285, which is lower by comparison with the other algorithms tested.
- To predict the deflection, we recommend using the SVR algorithm, which has an MSE score of 0.1417.
- To identify if the part is a good part or a defective part most of the algorithms tested have the same accuracy (64%). Hence, either LDA, ANN, KNN, SVC or Logistic Regression can be used to identify the quality of the part.
- To identify the overall shape, a CNN model was used, which delivered a model with 92.5% accuracy.
- To determine the quality of transparency, a CNN model was used, resulting in a model with 75% accuracy, which may have some overfitting.

For future improvements, we would like to collect more data by printing more parts, collecting the real time data of the machine, performing more tensile tests and

calculating the deflection for more parts. Also, we would like to investigate a better way to classify the transparency of the circle in the part.

Appendix A: Raw Dataset

Red Parts:

	Pressure	Hold Pressure	Temp(L=230,R=240,H=260)	Label	thick	width	Tensile Strength (MPa)	Deflection(185g)
1	65	56	240	Red	2.912	12.54	25	42.8
2	65	56	240	Red	2.886	12.63	24.81	42
6	65	56	240	Red	2.925	12.86	24.27	42.6
7	62	56	240	Red	2.918	12.57	24.89	43
8	62	56	240	Red	2.943	12.54	24.61	
9	62	56	240	Red	2.916	12.67	24.94	43.2
10	62	56	240	Red				
11	62	56	240	Red	2.919	12.79	24.2	43
12	68	56	240	Red	2.836	12.56	25.61	43
13	68	56	240	Red	2.91	12.6	24.61	42.6

14	68	56	240	Red	2.92	12.6	24.75	
15	68	56	240	Red	2.887	12.61	24.94	42.6
16	68	56	240	Red	2.928	12.58	24.94	44
17	55	56	240	Red	3.085	12.73	24.91	41.6
18	45	40	240	Red	3.069	12.91	24.29	42.4
19	45	40	240	Red	3.083	12.79	24.09	42.2
20	45	40	240	Red	3.103	12.91	24.29	42
21	45	40	240	Red	3.077	12.57	24.94	42.8
22	42	40	240	Red	3.111	12.59	25.15	42
25	42	40	240	Red	3.099	12.61	25.01	42.2
26	42	40	240	Red	3.057	12.57	24.97	42.2
27	42	40	240	Red	3.092	12.64		42
29	48	40	240	Red	3.046	12.73	24.25	42.6
31	48	40	240	Red	3.072	12.57	25.45	43
40	45	40	240	Red	2.98	12.63	26.27	42.6
3	65	56	240	Red				
4	65	56	240	Red				

5	65	56	240	Red				
23	42	40	240	Red				
28	48	49	240	Red				
30	48	40	240	Red				
						avg	24.83434783	42.56363636
						min	24.09	41.6
						max	26.27	44

		Average		Min		Max		
		Tensile Strength	Max Deflection	Tensile Strength	Max Deflection	Tensile Strength	Max Deflection	
	Red	24.83434783	2.563636364	24.09	1.6	26.27	4	

	Green	24.485	2.628571429	23.96	2	25.08	3	
	Good	24.5392 1053	1.989189189	23.62	1.2	25.55	2.8	
	Good part stays between Red and Green by Tensile Strength							
	Good part has lowest Deflection							

Green Parts:

ID	Pressure	Hold Pressure	Temp(L=230,R=240,H=260)	Label	thick	width	Tensile test	Deflection (185g)
5	55	50	230	Green	2.939	12.58	25.08	43
11	48	45	260	Green	2.975	12.59	23.96	
12	48	45	260	Green	2.955	12.58	24.17	43
13	48	45	260	Green	2.92	12.61	24.49	
14	48	45	260	Green	2.92	12.6	24.75	43
15	48	45	260	Green	2.93	24.46	24.46	43
16	48	50	260	Green	2.924	12.55		42.2
17	48	45	260	Green				42
18	48	45	260	Green	2.924	12.55		42.2

1	55	50	230	Green				
2	55	50	230	Green				
3	55	50	230	Green				
6	55	50	230	Green				
7	55	50	230	Green				
8	55	50	230	Green				
8	48	50	260	Green				
9	55	50	230	Green				
9	48	50	260	Green				
10	55	50	230	Green				
						avg	24.485	42.62857143
						min	23.96	42
						max	25.08	43

Red/Green Parts

ID	Pressure	Hold Pressure	Temp(L=230,R=240,H=260)	Label	thick	width	Tensile test name
1	55	40	260	Red/Green			
2	55	40	260	Red/Green			

3	55	50	260	Red/Green			
4	55	50	230	Red/Green			
4	55	50	260	Red/Green			
5	55	50	260	Red/Green			
6	52	50	260	Red/Green			
7	52	50	260	Red/Green			
11	65	50	230	Red/Green			
12	65	50	230	Red/Green			
13	65	50	230	Red/Green			
14	65	50	230	Red/Green			
15	65	50	230	Red/Green			
16	48	50	260	Red/Green			
16	65	50	230	Red/Green			
17	68	50	230	Red/Green			
18	68	50	230	Red/Green			
19	60	50	260	Red/Green			
19	68	50	230	Red/Green			
20	68	50	230	Red/Green			
21	60	50	260	Red/Green			
21	68	50	230	Red/Green			
22	45	40	230	Red/Green			
22	60	50	260	Red/Green			

23	60	50	260	Red/Green			
23	45	40	230	Red/Green			
24	45	40	260	Red/Green			
24	45	40	230	Red/Green			
25	45	40	230	Red/Green			
25	45	40	260	Red/Green			
26	45	40	230	Red/Green			
26	42	40	260	Red/Green			
26	60	50	260	Red/Green			
27	48	40	230	Red/Green			
27	42	40	260	Red/Green			
28	48	40	230	Red/Green	3.058	13.04	
28	38	35	260	Red/Green			
29	48	40	230	Red/Green	3.036	12.81	
29	38	35	260	Red/Green			
30	38	35	260	Red/Green			
30	48	40	230	Red/Green			
31	48	40	230	Red/Green			
31	38	35	260	Red/Green			
32	38	35	260	Red/Green			
33	38	35	260	Red/Green			
34	38	35	260	Red/Green			

35	38	35	260	Red/Green			
36	38	35	260	Red/Green			
37	38	35	260	Red/Green			

Good Parts:

ID	Pressure(front)	Pressure(back)	Temp	Label	thick	width	Tensile Strength (MPa)	Deflection
1		56	240		2.993	12.6	24.22	
2			240		2.965	12.61		
3			240		3.002	12.61	23.69	41.8
4			240		2.993	12.68	23.76	42
5			240		2.963	12.67	24.42	42.2
6			240		2.954	12.57	25.55	42.6
7			240		2.958	12.66	24.76	42
8			240		2.961	12.54	24.58	42
9			240					
10			240					42.8
11			240					

12			240					
13			240					
14			240		2.994	12.59	24.7	
15			240		2.963	12.58	25.36	41.6
16			240		2.979	12.55		42.2
17			240					42
18			240					
19			240					42
20			240					
21			240					
22			240					
23			240					
24	56		240		2.939	12.6	24.74	
25			240					
26			240					
27			240					
28			240		2.967	12.58	24.9	41.8

29			240					42
30			240		3.1	12.63	23.62	42.2
31			240					
32			240					
33			240					
34			240		2.954	12.69	24.05	42
35			240					
36			240		2.941	12.59	24.97	42.2
37			240		2.932	12.58	24.87	41.2
38			240		3.028	12.56	24.31	41.8
39			240		2.968	12.63	24.38	
40			240		2.975	12.56	24.58	42.2
41			240					
42			240					
43			240					
44			240					
45			240		2.952	12.57	24.6	41.6

46			240		2.974	12.57	24.85	42
47			240				24.55	42.2
48			240		2.942	12.59	24.75	41.8
49			240					
50			240					
51			240					
52			240					
53			240					
54			240					
55			240					
56			240					
57			240					
58			240		2.968	12.61	24.48	42.2
59			240					
60			240					
61			240					
62			240		2.95	12.57	24.53	

63			240		2.937	12.56	24.41	41.8
64			240		2.956	12.58	24.49	41.8
65			240		2.957	12.56		42
66			240		2.974	12.56	24.13	41.8
67			240		2.931	12.59		
68			240					
69			240					
70			240					
71			240					
72			240					
73			240					
74			240		3.022	12.56	24.29	42
75			240		2.949	12.57	24.49	42.6
76			240		2.931	12.56	24.61	
77			240		2.991	12.53	24.56	42
78			240					
79			240					

80			240					
81			240					
82			240					
83			240					
84			240					
85			240					
86			240					
87			240					
88			240					
89			240					
90			240		2.952	12.56	24.87	41.8
91			240					
92			240					
93			240					
94			240					
95			240					
96			240					

97			240					
98			240					
99			240					
100			240					
101			240					
102			240					
103			240		2.928	12.57	24.67	41.8
104			240		2.966	12.55	23.85	
105			240		2.933	12.58	25	41.6
106			240		2.953	12.59	25.03	42.2
107			240		2.955	12.58	24.33	41.8
108			240					
109			240					
110			240					
111			240					
112			240					
113			240					

114			240					
115			240		2.966	12.54	24.54	42
116			240					
117			240					
118			240					
119			240					
120			240					
121			240					
122			240					
123	55		240					
124			240					
						avg	24.53921053	41.98918919
						min	23.62	41.2
						max	25.55	42.8

Appendix B: Statistics on the Pressure and Temperature of the Parts

Attribute	Red Label (Pressure Varied)	Green Label (Temperature Varied)	Red/Green Label (Pressure and Temperature Varied)
Avg. Temp.	240 C	245.79 C	246 C
Temp. Std.Dev	0	14.98 C	14.92 C
Temp. Range	240 C	230-260 C	230,260 C
Avg. Pressure	50.96 MPa	51.3 MPa	51.6 MPa
Pressure Std.Dev	8.731 MPa	3.5 MPa	10.36 MPa
Pressure Range	45-68 MPa	48-55 MPa	38-68 MPa
Avg. Holding Pressure	49MPa	48 MPa	43.7 MPa
Holding Pressure Std.Dev	7.8MPa	2.4 MPa	6.3 MPa
Holding Pressure Range	40-56	45-50 MPa	35-50 MPa

Appendix C: Statistics on the Tensile Test Data

Data Features for Tensile Strength:

Red Label (Pressure Varied)

- Average: 24.8 MPa
- Standard Deviation: 0.49 MPa
- Range: 24.09-26.27 MPa

Green Label (Temperature Varied)

- Average: 24.485 MPa
- Standard Deviation: 0.365 MPa
- Range: 23.96-25.08 MPa

Good parts

- Average: 24.54 MPa
- Standard Deviation: 0.408 MPa
- Range: 23.62-25.55 MPa

Appendix D: Detailed Deep Learning Architecture

ANN - Regression (Tensile Strength - Epoch:100)

Layers	Size	Activation Function
Output	1	None
Hidden 2	8	ReLU
Hidden 1	64	ReLU

ANN - Regression (Deflection - Epoch:300)

Layers	Size	Activation Function
Output	1	None
Hidden 2	12	ReLU
Hidden 1	64	ReLU

ANN - Classifier (Epoch:150)

Layers	Size	Activation Function
Output	1	Sigmoid
Hidden 2	12	ReLU
Hidden 1	64	ReLU

CNN - Overall Shape (Epoch:60)

Layers	Filters	Size	Stride	Activation Function
Output	1			Sigmoid
Fully Connect	16			ReLU
Fully Connect	64			ReLU
Fully	256			ReLU

Connect				
Conv2D	128	2		ReLU
MaxPool		2	1	
Conv2D	64	3		ReLU
AvgPool		3	1	
Conv2D	32	3		ReLU

CNN - Transparency (Epoch:50)

Layers	Filters	Size	Stride	Activation Function
Output	1			Sigmoid
Fully Connect	256			ReLU
MaxPool		2	1	
Conv2D	32	3		ReLU
MaxPool		2	1	
Conv2D	32	3		ReLU