



Bilkent University

Department of Computer Engineering

## Senior Design Project

Machine Learning for Machining Processes of  
Three-Dimensional Parts

## Project Low Level Design Report

**Project Members:** Irmak Akyeli, Denizhan Kemeröz, Alp Üneri, Bulut Gözübüyük, Tuva Tanay Işıksal

**Supervisor:** Prof. Dr. Uğur Güdükbay

**Jury Members and Project Evaluators:** Dr. Shervin Arashloo and Dr. Hamdi Dibeklioglu.

Feb 26, 2022

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

# Contents

<b>1. Introduction</b>	<b>2</b>
1.1 Object Design Trade-Offs	3
1.2 Interface Documentation Guidelines	5
1.3 Engineering Standards	9
1.4 Definitions, Acronyms and Abbreviations	9
1.4.1 Artificial Intelligence (AI)	9
1.4.2 Machine Learning (ML)	9
1.4.3 Machining Process Identification (MPI)	9
1.4.4 Convolutional Neural Networks (CNN)	10
<b>2. Packages, Classes &amp; Libraries</b>	<b>10</b>
<b>3. Machine Learning Model</b>	<b>12</b>
<b>4. Validation and Testing</b>	<b>14</b>
<b>5. References</b>	<b>17</b>

# 1. Introduction

The project will be an application of machine learning (ML) techniques in the field of machining process identification (MPI). ML is defined as a branch of artificial intelligence (AI) and computer science that focuses on the use of data sets and algorithms to mimic the way humans are able to learn on computer systems [1]. ML algorithms, often called models, are fed data from data sets that gradually get better and better in their accuracy in the same ways that humans would [1]. For example, an ML model may be constructed to identify how many faces are in a picture that is supplied. In order to do this, the model would have to be trained using a known data set, which involves feeding the model data on which the aspect in question is known, and the model would then gradually get better and better at identifying how many faces there are in a supplied picture. After sufficient training, the model would be able to identify the number of faces within a picture with great confidence.

MPI using ML is a novel area of research that aims to automate the manufacturing of mechanical parts by automatically deciding whether the part in question would be able to be manufactured using the manufacturing plants in place in an effort to make the manufacturing process more time and cost-effective and to reduce the number of faults that may occur during manufacturing [2]. Our project will investigate the applicability of ML and especially convolutional neural networks (CNNs) for MPI means. A CNN is a deep learning model that takes in an input image and by assigning importance such as learnable weights and biases to various aspects or objects within the image is able to differentiate images within a number of categories the aspects/objects within the image can belong to [3].

We will be trying to come up with an ML model that will be used to automate the processes of determining the type of manufacturing processes (additive versus subtractive manufacturing) to be used, the producibility of three-dimensional models, and cost

estimation. Additive manufacturing is when processes build objects by adding materials layer by layer to form the desired product, while subtractive manufacturing is when materials are removed from an already existing object to come up with the part that is needed [4]. We are planning on developing a deep learning framework for determining which type of machining processes are suitable for producing a machine part and the producibility of these parts using the selected machining process for the provided three-dimensional models.

The machining operations we will consider are turning, milling, and drilling. Turning is a machining operation in which a workpiece is rotated while the cutting piece moves in a linear motion to achieve the desired shape, which is often cylindrical [5]. A lathe machine is usually used for turning purposes [5]. Drilling is a machining operation in which a drill press or a tapping machine is used to create a round hole in a workpiece [5]. Lastly, milling is a machining operation that involves using many multi-point rotary cutters to remove material from an object to achieve the desired outcome [5]. Our end product will also be able to determine which operations are to be performed in order to attain the desired part if the part is able to be produced using the available producing plant. Another stage for the parts that are producible using the selected machining process and the operations selected is the estimation of the cost of the production of the parts for different material properties and costs input to the system.

## 1.1 Object Design Trade-Offs

The four main factors that we deliberated on are algorithmic efficiency, memory cost, the accuracy of the model, and the time required to predict the attributes for the parts. The first two are common to all software engineering solutions, and we have decided that we would be in favor of algorithmic efficiency as opposed to lowering memory cost, and would make trade-offs in order to speed up our processes even when it means more memory would

be required. We have chosen to do so because our training of the model takes quite a bit of time with a large number of epochs and a sizable amount of time per each epoch. Therefore, we have chosen to try and speed this process up as much as we can even though it might mean we would have to utilize more memory to do so. One can argue that because we only need to train the model once before providing it to the user we would not necessarily have to worry about the time it would take to complete the training of the model. However, as during testing we might change qualities about the model in an effort to increase its accuracy, and would have to train it every single time, we have chosen to favor algorithmic efficiency as opposed to a lowered memory cost. One may also argue that since the training has to only be done once the memory cost will only have to be paid once, so it is not a big problem to favor algorithmic efficiency.

The other two factors we considered are the accuracy of the model when predicting a part, and the time it takes for the model to predict a part. We would ideally like the accuracy of the model to be as high as possible, and the time it takes to predict a model to be as low as possible. We have decided that the accuracy threshold for our model in order to be considered a success is 95%, and we have decided that it can take up to a minute's time for the model to predict a part that is given, depending on the size of the part.

## 1.2 Interface Documentation Guidelines

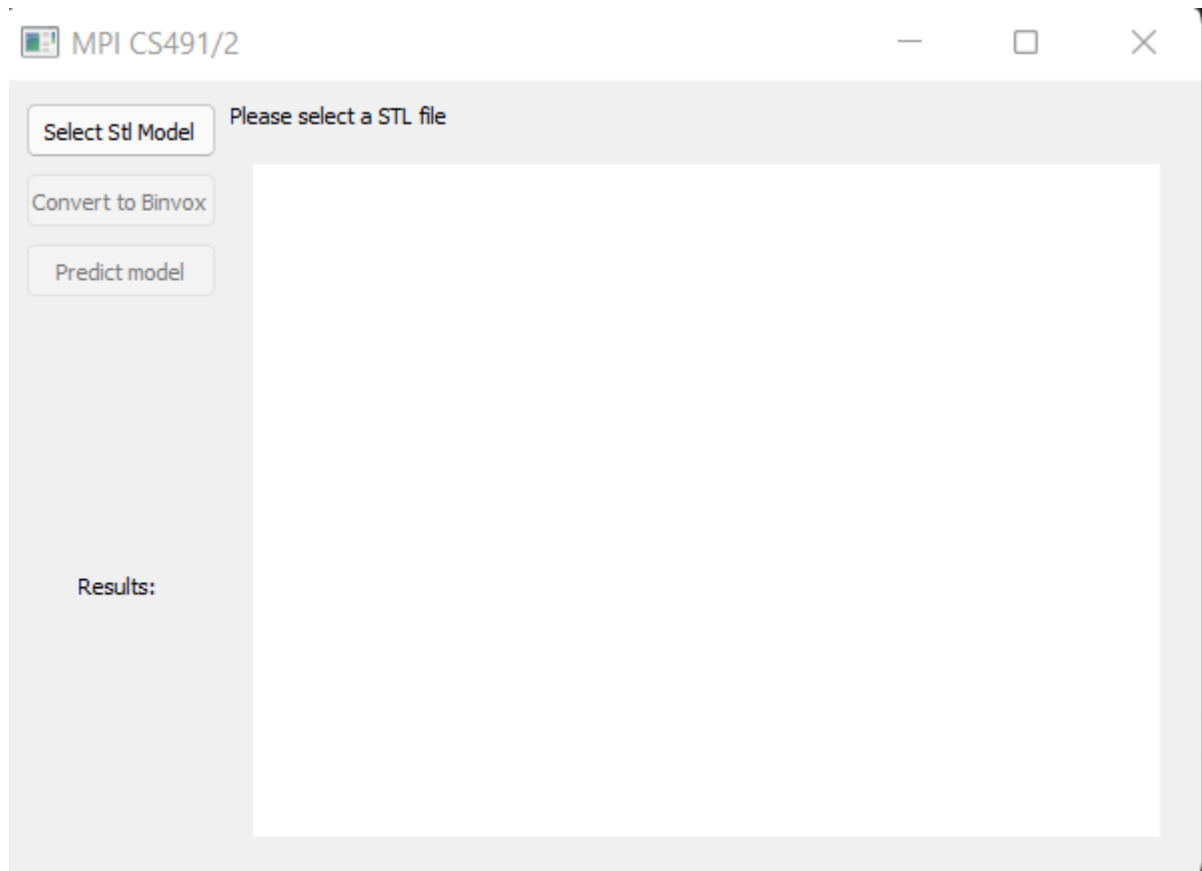


Figure - 1

Upon launching our program, the user will be met with the above screen. From this screen, only the Select STL Model button will be available. Using this button, the user will be able to select the STL model of the part that they are wanting to predict.

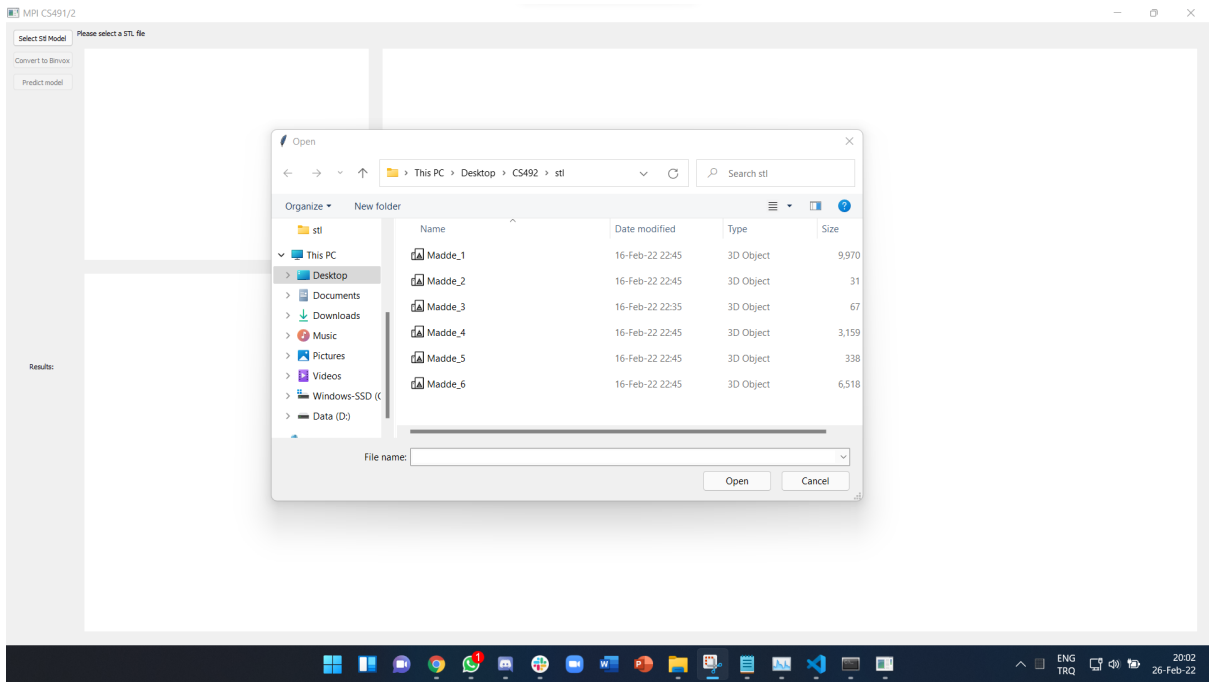


Figure - 2

Once they click the Select STL Model button, the user will be met with the above standard file selecting screen. From this screen they will be able to navigate their files and select the desired STL model.

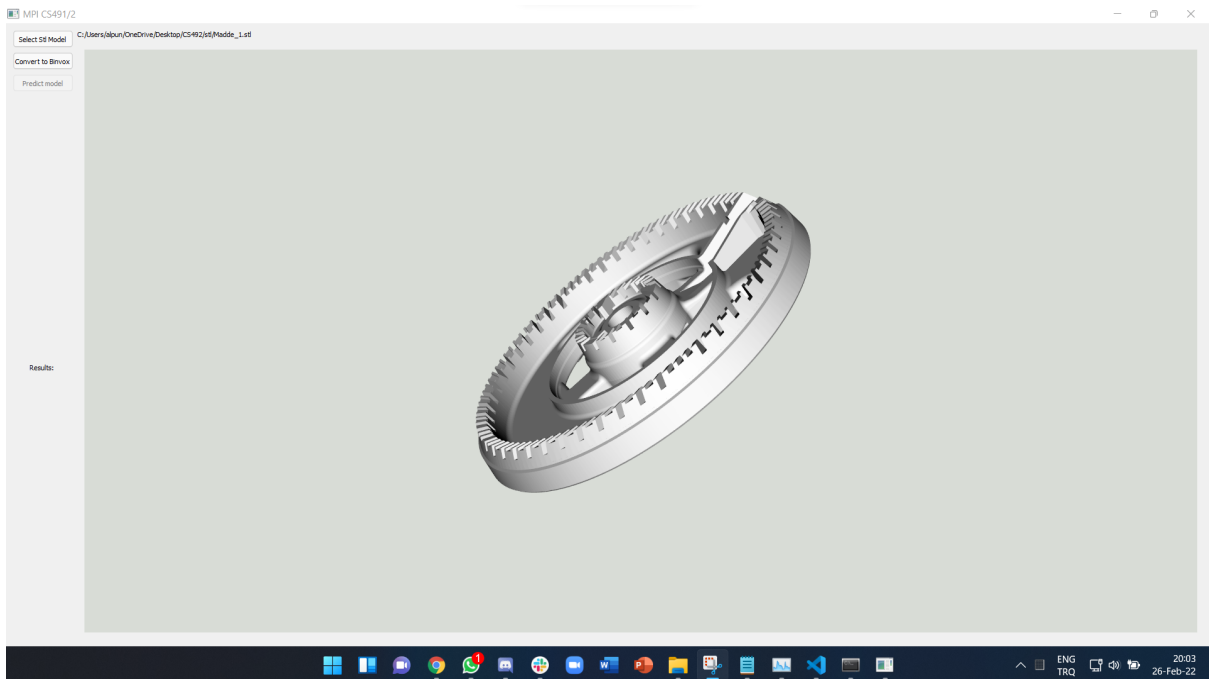


Figure - 3

After the user selects the STL model that they wish to predict, the model will be visible in the model viewer part of the screen. Here the STL model will be shown in 3-D and the user will be able to rotate or zoom in/out of the model as they wish. Also, the Convert to BinVox button will be available once the user has selected an STL model.

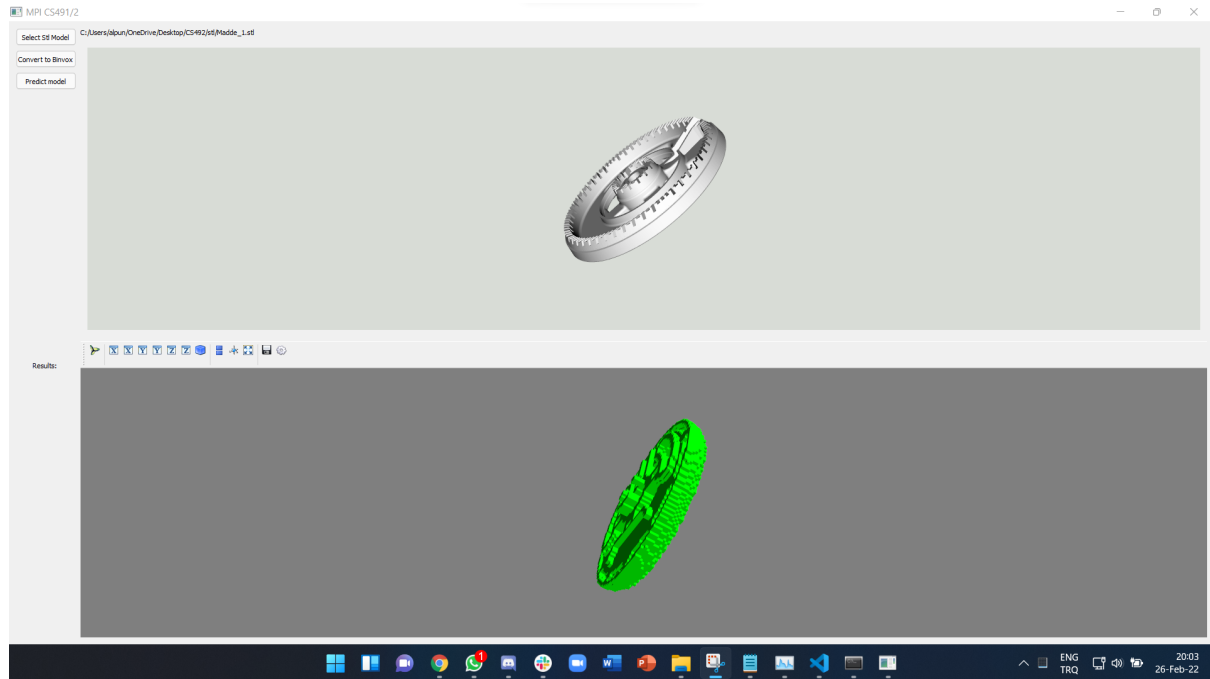


Figure - 4

After clicking the Convert to BinVox button, the supplied STL model will be converted into BinVox format and will then be displayed under the STL model. The user will likewise be able to rotate and zoom in/out to this model as well. After the user has converted the STL model into BinVox, the Predict Model button will become enabled as well.



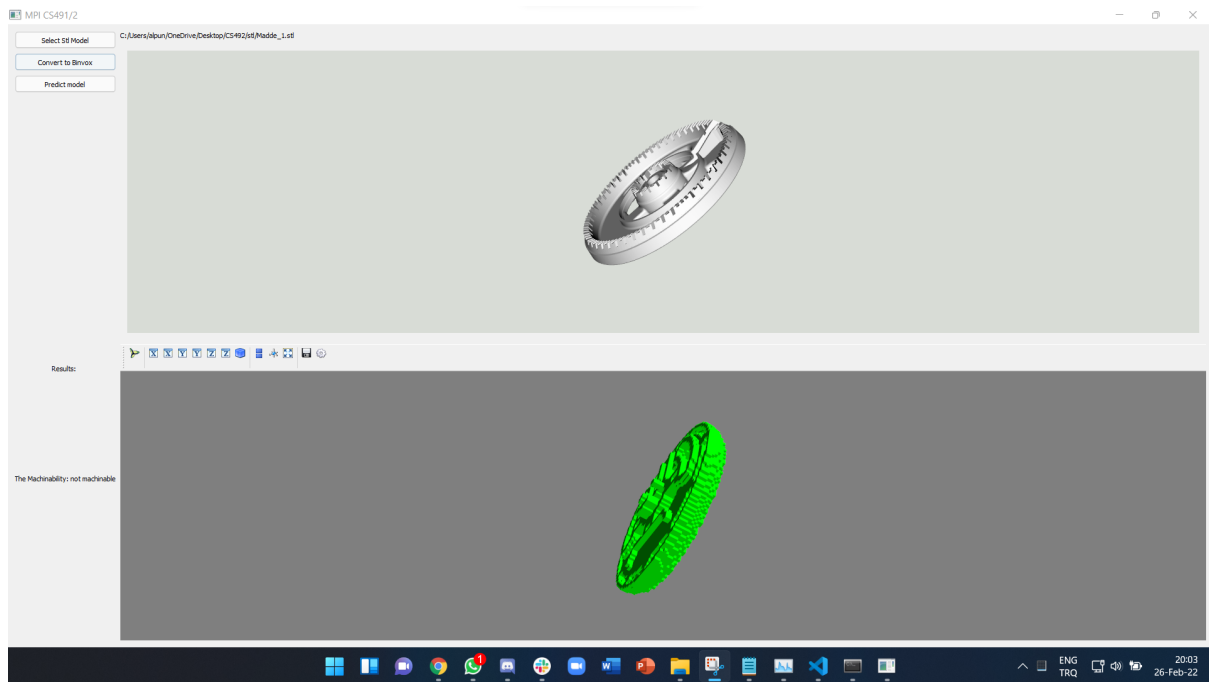


Figure - 5

Once the user clicks the Predict Model button, the model will be predicted and the results will be shown on the lower left hand corner of the screen. For this particular model the prediction results are that it is not machinable, so the machining operations are not shown. If the model is predicted to be machinable, the appropriate machining operations will also be shown as in the following screen.

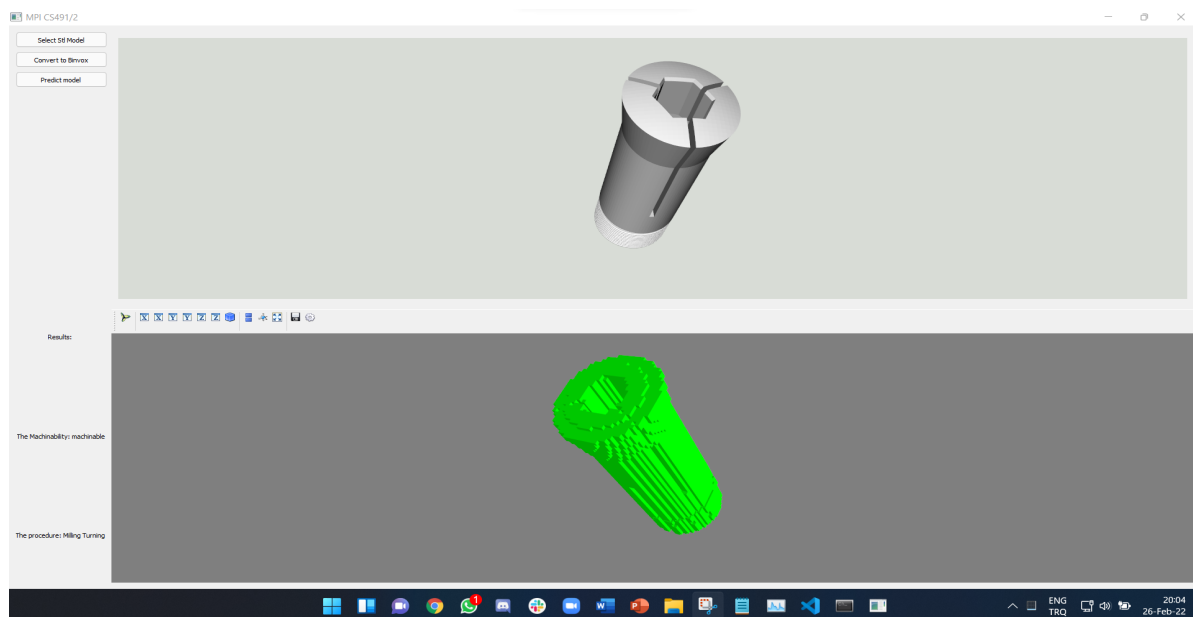


Figure - 6

As in the above screen, if the given model is predicted to be machinable, the machining operations will be shown in the lower left hand corner of the screen as well.

## 1.3 Engineering Standards

We have chosen to use the Unified Modeling Language (UML) when preparing diagrams for our system, and we have made a use-case diagram, object and class diagram, state machine diagram, sequence diagram, and activity diagram for our system. These diagrams can be found in our High Level Design Report.

## 1.4 Definitions, Acronyms and Abbreviations

In this section of the report we will be providing the definitions of acronyms and abbreviations that we will be using throughout the report.

### 1.4.1 Artificial Intelligence (AI)

Artificial Intelligence is a branch of computer science that is concerned with building smart machines capable of performing tasks that typically require human intelligence.

### 1.4.2 Machine Learning (ML)

Machine learning is defined as a branch of AI and computer science that focuses on the use of data sets and algorithms to mimic the way humans are able to learn on computer systems [1].

### 1.4.3 Machining Process Identification (MPI)

Machining process identification is the operation of deciding whether the part in question would be able to be manufactured using the manufacturing plants in place.

Augmenting MPI with ML is a novel area of research performed in an effort to make the manufacturing process more time and cost-effective and to reduce the number of faults that may occur during manufacturing [2].

#### 1.4.4 Convolutional Neural Networks (CNN)

A CNN is a deep learning model that takes in an input image and by assigning importance such as learnable weights and biases to various aspects or objects within the image is able to differentiate images within a number of categories the aspects/objects within the image can belong to [3]. There are different dimensional CNNs, such as two dimensional and three dimensional CNNs. We will be using a three dimensional CNN throughout our project.

## 2. Packages, Classes & Libraries

As we have explained before, our project is not an object oriented project. Therefore we do not use components like interfaces and abstract classes. Instead, we have a main class, a prediction class and a training class. From the nature of our project, the main class accesses the predict class when the user clicks the predict button. The training class is not linked to any other classes and is not accessed by them. Its sole purpose is to create trained CNN models that we will provide to the customer and the class itself will not be provided in the source code. Having only 3 classes the implementation packages were not necessary.

In our project however, we have a 3D CNN model training system that uses many different libraries provided by Python that should be mentioned to understand the training process. A brief summary of the libraries used and their descriptions are as follows:

- Numpy: Numpy is a powerful python library that enables the user to work and operate on multidimensional arrays. In our project numpy is used to store binvox files

transformed to multidimensional arrays and is the input type of our CNN training models.

- **Tensorflow:** Tensorflow is an open-source python library for ML and AI. It is an easy to use tool that lets the user create many different ML models such as deep-learning models like CNN. Keras is also a specialized version of tensorflow that focuses on deep-learning.
- **Keras:** Keras is an interface implementation for tensorflow that simplifies the use of the library in order to achieve better results and understanding. In our project the implementation of our CNN models are made with tensorflow, keras. From Keras we are using layers: Input, MaxPooling3D, MaxPooling2D, Dense, Flatten, BatchNormalization, Dropout and Conv3D. As well as its optimizers and regularizers.
- **Pandas:** Pandas is a python library created for data manipulation and analysis. In our project pandas is used to create training and validation partitions of our data.
- **Binvoy\_rw:** Is an open source code made for python that enables the user to work on binvox files and manipulate them to create different variable types. In our project as the name suggests binvox\_rw is used to manipulate our binvox files and turn them to numpy arrays.

### 3. Machine Learning Model

Like mentioned in the previous reports about our project, our aim is to improve the accuracy of our source code, add aspects missing in it and turn it to a working MPI system. To do this we need to analyze and understand the training model provided. We currently have three 3D CNN networks for three different tasks. The first one takes 21 different feature used in machining and categorize them under turning, milling and non-machinable features, the second one takes a machinable data input and predicts its machining procedure that can be milling, turning or both of them and finally the third model takes a data input and decides if it is machinable or not. The first CNN network, as you can see in Figure 7, operates on 200 distinct part models that are each taken from 6 different orientations for each feature. This makes a total of 20.000 models that are used in the creation of the CNN network. 70% of this data is used in the training, 15% of it in the validation and the remaining 15% is used in the testing. The CNN model is created using keras's sequential model, a model used for layering with single input and output of each layer, and has 10 layers. As the results of the other two models are dependent on the features provided, this CNN network is the most detailed one and is trained on 20,000 epochs. The Second and third CNN network, as you can see in Figure 8, is initialized with the weights loaded from the first network (the weights obtained after the training on 20,000 epochs) so that they are not required to train from the beginning. The second CNN network is fed 500 milling, 500 turning and 500 milling-turning models again all taken from 6 orientations. With the weights already loaded from the first network, only fine tuning is performed during the training and 500 epochs is used to achieve this. The last network takes the input data of the second network as machinable models and takes another 2400 models as non-machinable models. Again similar to the second network the weights are loaded from the first network and only 500 epochs are used to train the network. All the three networks show over 95% accuracy on the training.

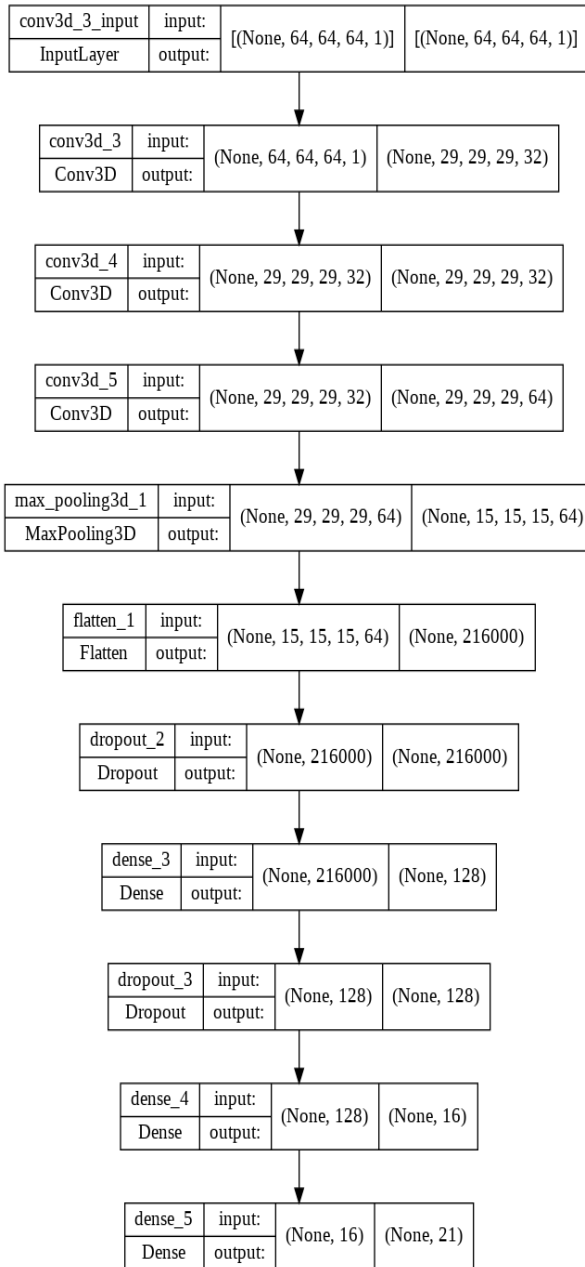


Figure - 7

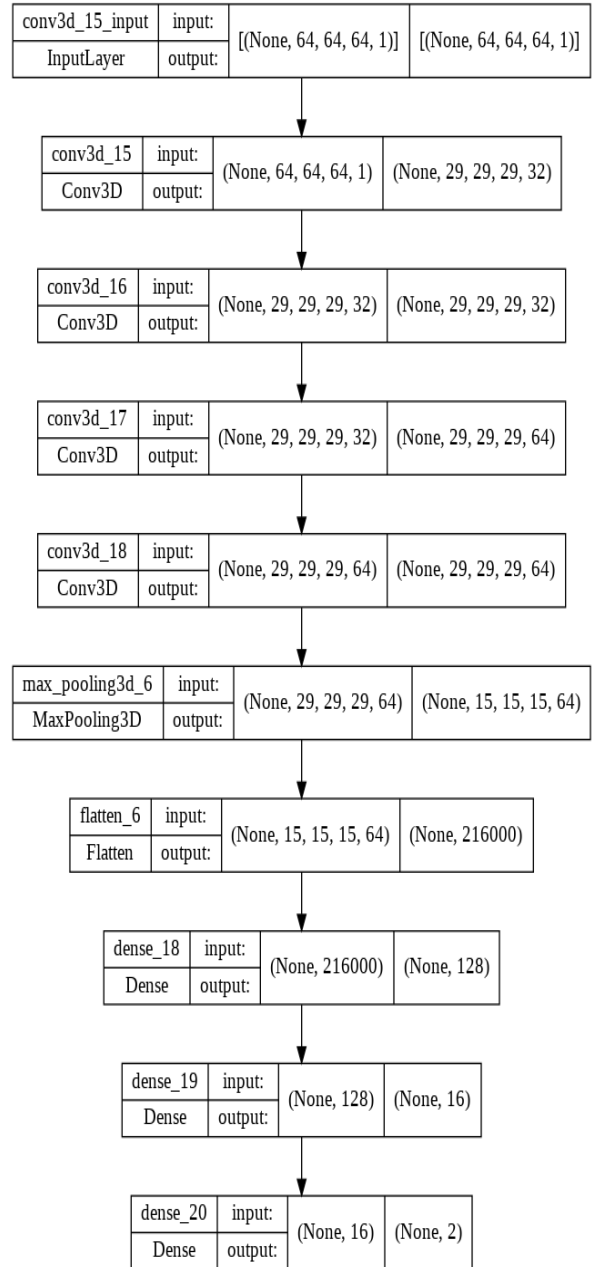


Figure - 8

## 4. Validation and Testing

However, contrary to the accuracies shown during the training outputs, when the model is tested on the real parts, concerning results are obtained and we are currently trying to improve them. The network shows a tendency to predict milling-turning for the machining parts that should be only turning and almost never predicts turning alone. Likewise, while looking at the machinability three out of 8 models returns wrong predictions and when the same binvox file is used to predict multiple times, the result also varies. All these makes the networks unreliable and in need of improvement. There are several possible reasons as of why the user receives these predictions:

- The data used in the training may not be enough for small enough details or accurate predictions.
- The resolution of binvox files that is currently 64x64x64 for competitive purposes is not enough for distinction of different shapes that we are trying to recognize as features, and tempers with the smooth corners too much to make them inseparable from sharp corners. Differences can be seen in figure 7 and 8.

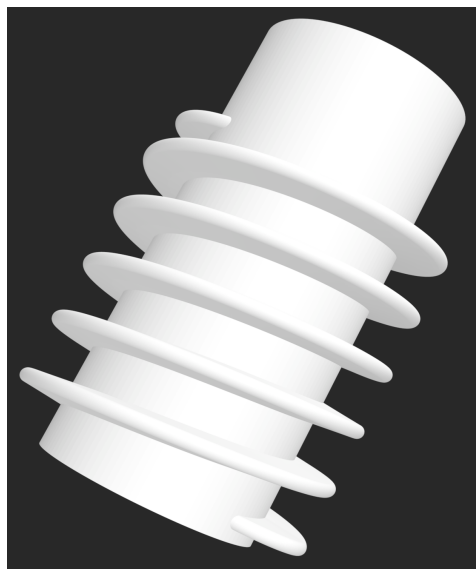


Figure - 9 ( 3D Stl Model)

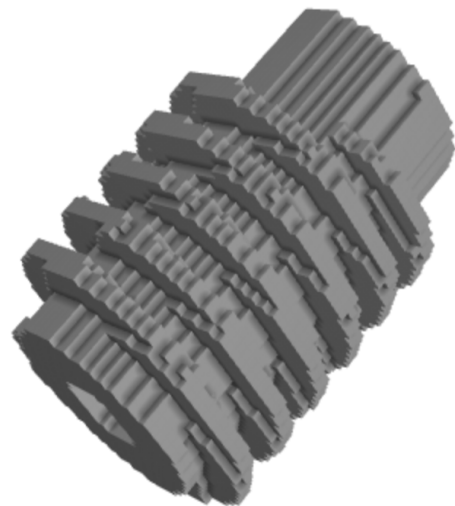


Figure - 10 (Voxel Format)

- The different numbers of features and models makes the CNN network biased towards a certain procedure. (There are 9 milling features compared to 7 turning features that can make the machine side with milling with unsure predictions)

- The training epochs choice and different trainings affect the output too much.
- The saving and preservation of the CNN model somewhat affects its prediction and results.

In our project we are currently seeking different approaches to improve the results of the predictions before adding new features to our system. Our current trials to improve the results include:

- Feature extraction, to make the number of features equal or block a particular feature that can affect the predictions in a negative way.
- Data extraction to make the number of models of machinable and nonmachinable equal to see if it makes the network biased.
- Use SHAP (SHapley Additive exPlanations) that is a game theoretic approach to explain the outputs of the machine learning systems, to see the weights assigned to different features and their effects on the predictions.
- Create a side project that takes binvox files and improves their details by increasing their resolutions and smoothing the round corners.
- Trials to train the networks with different epochs and parameters, to be able to compare them.

The results of these trials will determine the next steps we will be taking during our project.





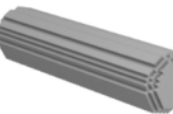

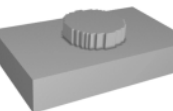
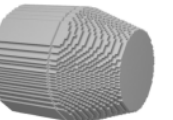


		ground truth		1.000 epochs		2.000 epochs		5.000 epochs		20.000 epochs		Data processed	
Model	#	is machinable?	procidure	is machinable?	procidure	is machinable?	procidure	is machinable?	procidure	is machinable?	procidure	is machinable?	procidure
	3	yes	milling	no	milling	no	milling	no	milling	no	milling	no	milling-turning
	19	yes	turning	no	milling-turning	yes	milling-turning	yes	milling-turning	yes	milling-turning	no	turning
	21	yes	turning	no	turning	no	milling-turning	no	milling-turning	no	turning	no	milling-turning
	30	yes	turning	no	milling-turning	no	milling-turning	no	turning	no	milling-turning	no	turning
	43	yes	milling	yes	milling	no	milling-turning	yes	milling-turning	yes	milling	no	milling-turning
	46	yes	turning	no	milling	no	milling	no	milling	yes	milling	no	milling
	49	yes	milling	yes	milling-turning	no	milling	no	milling	yes	milling-turning	no	milling-turning
	51	yes	milling	no	milling	no	milling	no	milling	yes	milling	no	milling
				2/8	4/8	1/8	1/8	2/8	4/8	5/8	4/8	0/8	3/8

Figure - 11 (Pink tone is wrong outputs and orange one is discussable)

## 5. References

- [1] “What is machine learning?”. <https://www.ibm.com/cloud/learn/machine-learning>.  
[Accessed: Oct 9, 2021]
- [2] S. G. Joung, V. Aggarwal, J. W. Sutherland, and M. B.-G. Jun, “Identifying manufacturability and machining processes using deep 3D convolutional networks,” *Journal of Manufacturing Processes*, vol. 64, pp. 1336–1348, 2021. M. Szilvsi-Nagy and G. Mátyási,
- [3] ”A Comprehensive Guide to Convolutional Neural Networks”.  
<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>. [Accessed: Oct 9, 2021]
- [4] “Additive vs. Subtractive Manufacturing”.  
<https://formlabs.com/blog/additive-manufacturing-vs-subtractive-manufacturing/>. [Accessed: Oct 9, 2021]
- [5] “Machining Processes: Turning, Milling, and Drilling”.  
<https://trimantec.com/blogs/t/machining-processes-overview>. [Accessed: Oct 9, 2021]

