Machine Learning Applications in CAD/CAM

A Seminar Report
submitted in partial fulfillment of the
requirements for the degree of
Master of Technology
by,

Sudhir Shinde
(173109009)
173109009@iitb.ac.in

Under the guidance of,

Prof. S.S. Pande

Department of Mechanical Engineering
Indian Institute of Technology, Bombay
Mumbai 400076 (India)

2019

DECLARATION

I certify that

- The work contained in the report is original and has been done by myself under the general supervision of my guide.
- The work has not been submitted to any other Institute for any credit, degree or diploma.
- I have followed the guidelines provided by the Institute in writing the report.
- I have confirmed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references.
- Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Sudhir Shinde

Acknowledgement

I would like to express my sincere gratitude to Prof. S. S. Pande, for all his encouragement, support and valuable guidance throughout the course of my work. I would also like to thank the members of the CAM Lab for their help and for creating an enjoyable working environment.

Sudhir Shinde
April 2019

Abstract

Machine learning is one of the most highly actively researched and sought after topic in recent times. There are areas in mechanical the field where we can apply Machine Learning to automate the process. One of the domain is CAD/CAM where we can use Machine Learning for object recognition, Feature extraction, 3D model reconstruction from 2D images, Defeaturing of CAD models for simulation. This report contains different applications of machine learning in CAD/CAM and current research challenges.
Keywords - CAD/CAM, Machine learning, Deep learning

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to Machine Learning

Machine Learning (ML) is a sub-field of computer science that evolved from the study of Pattern Recognition and Computational learning Theory in artificial intelligence.[1]
In simpler terms: It uses algorithms that iteratively learn from data and allows computers to discover patterns without being explicitly Programmed where to look. ML is nothing but an ability of Computers to learn from data or past experience. The data comes from various sources such as sensors, domain knowledge, experimental runs, etc [1]. It helps to make intelligent predictions or decisions based on the data.

Machine Learning refers to a broad family of algorithmic techniques which take advantage of historical data to learn behaviors, patterns, and functions to provide useful inference in a variety of scenarios. They attempt to learn by examples and are capable to capture complex relationships among collected data that are hard to describe, hence such methods are suitable for situations where physics-based modelling is not favourable to replicate behavior model [1]. Depending on the type of available data, learning can be performed in different ways as follows [1]:

- **Supervised (inductive) learning:** Training data includes desired outputs i.e., data are composed of input and the desired output is known. Such data is also called as labelled data.

- **Unsupervised learning:** Training data does not include desired outputs i.e. learning data are only composed of input. Such data is also called as unlabelled data. The data is analyzed and studied through clustering, reduce the dimensionality, etc into a different class.

- **Reinforcement learning:** It is about taking suitable action to maximize reward in a particular situation.

The rise of machine learning and deep learning is because of the availability of large data and computational power. The more powerful algorithm has been developed to train a large amount of data that lead to increased use of ML and DL [7]

1.2 Issues in AI and Machine Learning

Issues in ML

The main problem with Machine learning is that it requires extensive data set to train the algorithm for future prediction. Machine learning algorithms work on specific problems only, if there is any different test case then there are chances that the ML algorithm will fail [1].
It is not a guarantee that machine learning algorithms will always work in every case imaginable. Sometimes or most of the times machine learning will fail, thus it requires some understanding of the problem at hand in order to apply the right machine learning algorithm. Bias in the data has significant impact on the accuracy of the model [11].

Why ML?

Machine learning algorithm works better if we provide a large amount of data. As more data is available so we can train the ML algorithm more accurately. This large set of data requires large power for computation, more computation can be done easily by using GPU [11].
A lot of new algorithms have developed and many are at developing stage [11].

1.3 Applications of Machine Learning in CAD and CAM

With the increasing amount of available data, computing power, and network speed for a decreasing cost, the manufacturing industry is facing an unprecedented amount of data to process, understand and exploit. Phenomena such as Big Data, the Internet-of-Things, Closed-Loop Product Life cycle Management, and the advances of Smart Factories tend to produce humanly unmanageable quantities of data. This data can be helpful to build robust ML model.
In [8] paper they have mentioned how machine learning can be helpful for Surface Defect Detection. ANN and SVM algorithms are used to detect the surface defect

from the manufacturing part. SVM shows the better result on test data as compared to SVM. Jia et al. [8] tested system on image data collected directly from hot rolling processes. The training data have 920 images, and the testing data have 306 images. There are varieties of parts which are used for manufacturing assembly. Object detection is one step in ML that can be helpful for assembly of parts. Jia et al. proposed a real time Object Detection system with Region Proposal Networks. On the PASCAL VOC 2012 test set, they got an mAP(mean average precision) of 70.4%.
In Daito et al. [3] paper Deep Learning approach is used for predictive Diagnosis of machine failure. They have applied the convolution neural network to predict the different stages of machine failure. Fault diagnosis of ball bearings has been detected by Kankar et al. [9], they got 72% accuracy by using ANN. This can be further improved by using deep learning.

1.4 Scope of the Seminar

This seminar aims to study the various application of Machine Learning and Deep Learning in the CAD/CAM. The literature reviewed mainly focuses on CAD application such as Object recognition, Feature recognition, Defeaturing of CAD models. It aims to survey research directions for various applications in the CAD/CAM. Chapter 2 gives a detailed literature review with observations. Finally conclusive remarks are given in chapter 3.

Chapter 2

Literature Survey

2.1 Scope

The literature survey is divided into parts such as Overview of Machine Learning Techniques and the review of various technique applied to the CAD/CAM.

2.2 Overview of Machine Learning Techniques

With the increasing amount of available data, computing power and network speed for a decreasing cost, the manufacturing industry is facing an unprecedented amount of data to process, understand and exploit. Phenomena such as Big Data, the Internet-of-Things, Closed-Loop Product Life-cycle Management, and the advances of Smart Factories tend to produce humanly unmanageable quantities of data [5].

Machine learning (ML) refers to a broad family of algorithmic techniques which take advantage of historical data to learn behaviours, patterns and functions to provide useful inference in a variety of scenarios.

Various algorithms are used to train the data. List of various algorithms that are developed till date are given below -

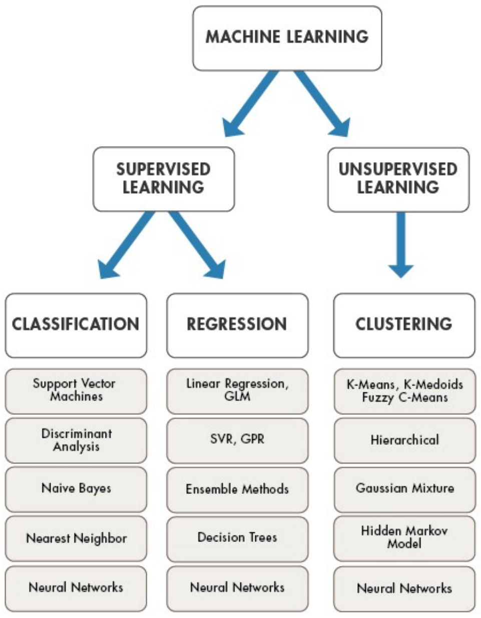


Figure 2.1: Classification of ML Algorithms [10]

Any machine learning algorithm is a hypothesis set which is taken before considering the training data and which is used for finding the optimal model. Machine learning algorithms have 3 broad categories -

- Supervised learning - the input features and the output labels are defined. If the data is labelled then we can use supervised learning e.g - Suppose you had a basket filled with fresh fruits. Your task is to classify the same type of fruits together. If you know the parameter of the fruits(color, shape) then the model will learn from the pre-classified data and predict on new unclassified fruits. This comes under supervised learning.

- Unsupervised learning - the data set is unlabeled and the goal is to discover hidden relationships. e.g - in the above example you know nothing about the fruits then to group the different fruits unsupervised learning is used.

- Reinforcement learning - some form of the feedback loop is present and there is a need to optimize some parameter.

The details of the important algorithm are given below -

- **Linear Regression -**

  With linear regression, the objective is to fit a line through the distribution which is nearest to most of the points in the training set. In simple linear regression, the regression line minimizes the sum of distances from the individual points, that is, the sum of the "Square of Residuals". Hence, this method is also called the "Ordinary Least Square". Linear regression can

also be achieved in case of multidimensional data i.e. data-sets that have multiple features. In this case, the 'line' is just a higher dimensional plane with dimensions 'N-1', N being the dimension of the data set [2].
e.g. Prediction of weather can be done by using linear regression.

- **Logistic Regression -**

  Logistic Regression although termed regression is a classification technique. Contrary to linear regression, logistic regression does not assume a linear relationship between the dependent and independent variables. Although a linear dependence on the logit of the independent variables is assumed [2].
  e.g. To classify an email as spam or not-spam logistic regression is used [1].

- **Support Vector Machines -**

  Support vector machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. In SVM, we plot the data points in an N-dimensional space where N is the number of features and find a hyper plane to differentiate the data points. This is a good algorithm when the number of dimensions is high with respect to the number of data points. Due to dealing with high dimensional spaces, this algorithm is computationally expensive [2].
  e.g. to classify the items into different category SVM is used.

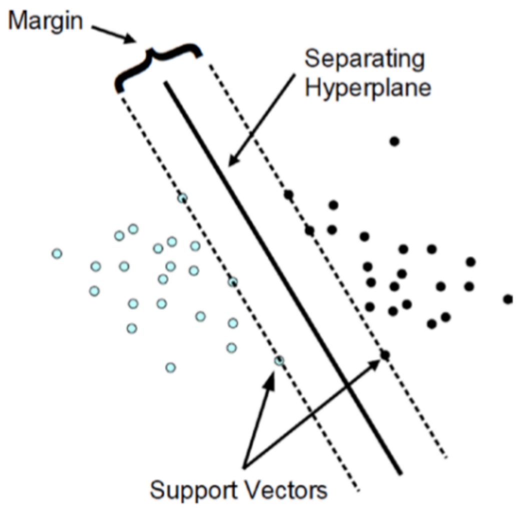


Figure 2.2: Classification using SVM [1]

- **K-means clustering -**

  k-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes a different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point, we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move any more [1]. Finally, this algorithm aims at minimizing an objective function known as squared error function given by:

$$J(V) = \sum_{n=1}^{c} \sum_{n=1}^{ci} ||xi - vj||^2$$

Where,
'$||xi - vj||'$ is the Euclidean distance between xi and vj.
'ci' is the number of data points in i^{th} cluster.
'c' is the number of cluster centers.

- **Decision Trees -**

  Decision tree algorithm falls under the category of the supervised learning. They can be used to solve both regression and classification problems.Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree [2].

- **Neural Networks -**

  Neural Network is a computational model that mimics the brain's processing to extract patterns out of data and act as universal function approximators. It is a machine learning model, where the neural network is trained on a set of training data, and validated on test data. The structure of a typical neural network is as given in Figure It consists of an input layer, which has several neurons. Then comes the hidden layers and finally there is an output layer. Each neuron in a layer is connected to the previous and the next layer, where each connection is given a certain weight. Various learning strategies such as supervised learning using gradient descent, backpropagation, etc are used to train the model [1].

  Neural Network has gained prominence in various applications such as computer vision, speech recognition, etc where its accuracy has improved significantly in the last few years. This has prompted research in applying neural network as a statistical tool in 3D shape analysis to extract relevant features and approximate it to the respective output.

Deep Learning involves the use of many hidden layers. This results in the extraction of higher levels of abstract features, that can help in approximating output [7].

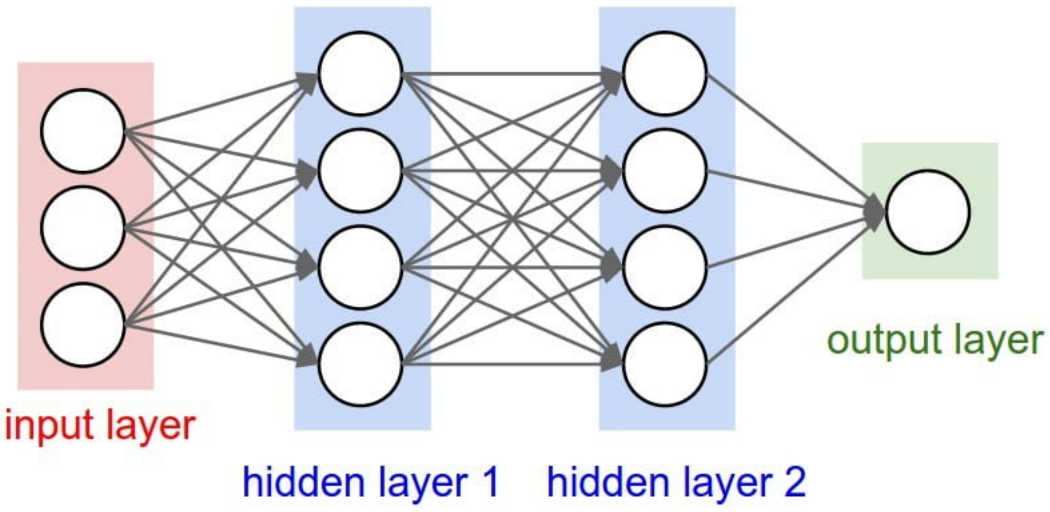


Figure 2.3: Structure of a fully connected Neural Network [11]

Accuracy of Neural Network is higher as compared to other machine learning algorithm for most of the application, followed by SVM which is good for classification problem. Again the accuracy of each ML algorithm is depends on the input data.

2.3 Object recognition and research challenges

The challenge of object recognition can be summarized as follows: “A vision system which makes use of an object model is referred to as a model-based vision system, and the general problem of identifying the desired object is referred to as object recognition. While there is no single definition of the object recognition problem, the objective is to identify a desired object in the scene and to determine its exact location and orientation.” [5]

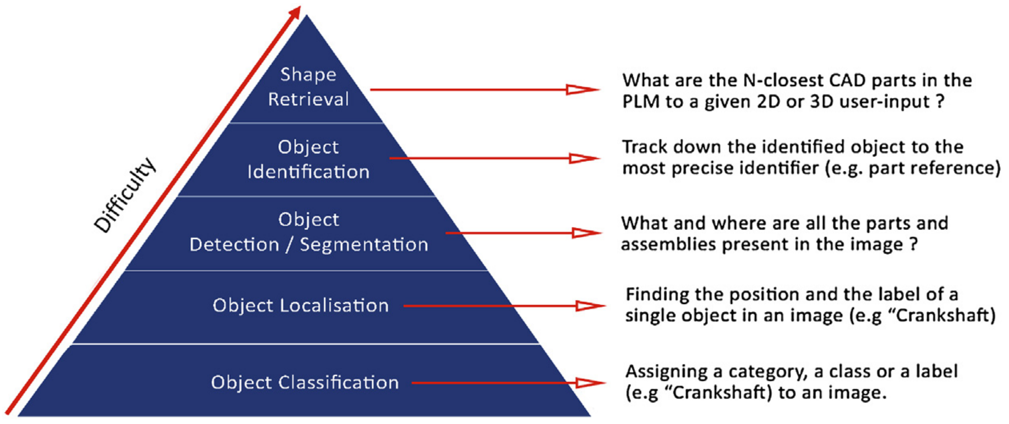

Difficulty
Shape Retrieval
Object Identification
Object Detection / Segmentation
Object Localisation
Object Classification
What are the N-closest CAD parts in the PLM to a given 2D or 3D user-input ?
Track down the identified object to the most precise identifier (e.g. part reference)
What and where are all the parts and assemblies present in the image ?
Finding the position and the label of a single object in an image (e.g "Crankshaft)
Assigning a category, a class or a label (e.g "Crankshaft) to an image.

Figure 2.4: Object recognition can be subdivided into five different tasks with an increasing difficulty [5]

Figure 2.4 presents five tasks in which the research area of Object Recognition can be subdivided. Each of the following tasks aims to address slightly different objectives with an increasing difficulty yet essential to overcome in order to specify and build a system able to perform cross-comparisons between 3D and 2D data. The major challenges in front CAD are dataset to train the model robustly. The reason to use or develop a standardized dataset is to enable research, to train the model robustly. The issues in object recognition tasks tend to make difficult to design such as background noise, different camera poses, deformation and damages, intraclass variation of different objects, noisy data etc.

2.4 Systems for Object Recognition

The upcoming sections will describe each step of the general work flow applied to object recognition given by Dekhtiar et al. [5] The Figure 2.5 given below will give the overall steps involved in object recognition

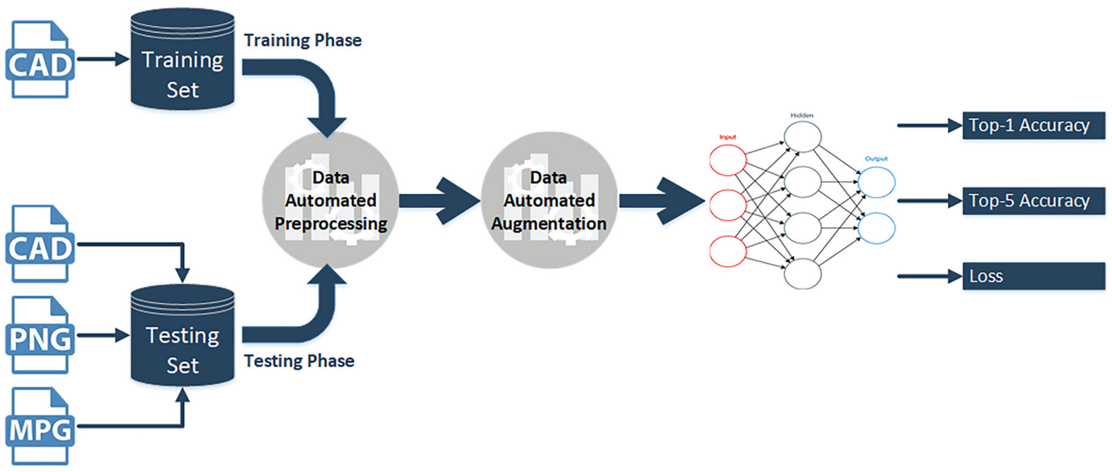


Figure 2.5: Workflow of Object Recognition [5]

2.4.1 Data preprocessing

No deep learning model handling STEP models has been published in the current state of the art and developing one from scratch would take too much time for a case-study. Nevertheless, some models can handle STL geometries; however, they do not adapt well to images. Nonetheless, images can be considered as a degraded view of geometry with only one viewpoint and no depth information. Thus, the type and format heterogeneity introduced by 3D models and images could be overcome by considering every CAD model as a series of images.

Figure 2.6 given below presents the strategy used to convert or translate each CAD model into N different images.

Each rotation has been decomposed into a precise number of steps. Different configurations corresponding to various strength of preprocessing have been tested, each of them generates a defined number of automatically taken screenshots.

Example Level: [1, α, β, γ, 1]

The number of parameters in each level corresponds to the number of steps in which the R2 rotation will be decomposed. For instance, this example level takes n $rot2 = 5$ different parameters and thus R2 will be divided into identical sub-rotations or steps equal to Performing n Is necessary to omit the initial camera position in the counting process. During each step of the R2 rotation, a full R1 rotation is performed. R1 is also divided into a fixed number of steps, n rot1. This number is defined by the value of the parameter corresponding to the current R2-step and thus change over time. Each R1-step have an angle equal to Each time a rotation along R1 is performed, a screenshot is taken.

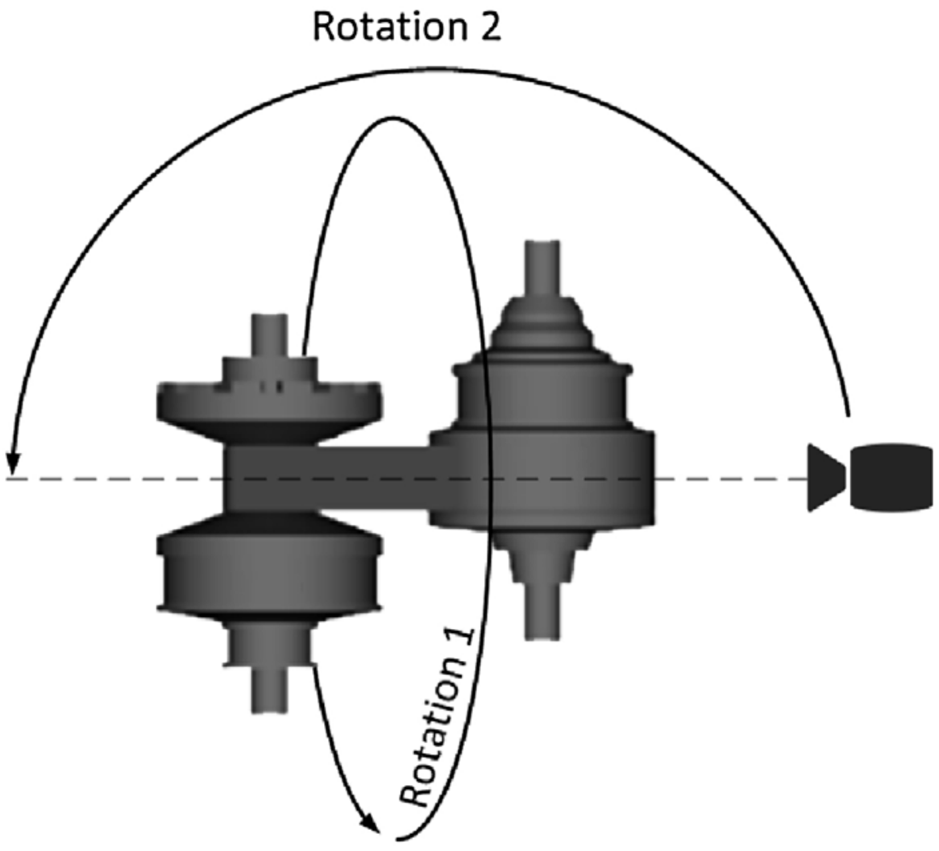


Figure 2.6: Translation each CAD model into N different images [5]

Dekhtiar et al. [5] has taken as many as 6, 12, 28, 38, 52 screenshots per CAD Model. It seems natural that the more screenshots are taken the less information is lost in the data preparation process. However, this situation presents a significant trade-off. With an increasing data preparation strength, an increasing number of viewpoints will be generated and thus seen by the deep learning model during the training phase. However, the more screenshots to process, the longer the training phase will be. Moreover, the amount of additional information brought by a higher strength in preprocessing converges toward zero. This is because close screenshots will start looking too similar and thus, a lot of redundancy will be introduced in the data which would not improve the final performances of the model and could only lead to longer computation times.

After testing each configuration, 38 screenshots per CAD model, corresponding to a “High Strength” preprocessing, has been chosen due to acceptable computation requirements and performances achieved by the final model.

- **Background -**
  A background is randomly added. The textures have been selected from the various other images. The objectives have been to add plausible backgrounds that could be found in a manufacturing context without being too restrictive. Thus, the following background families have been considered:

Bark, Bricks, Glass, Leather, Metal, Paper, Rock, Stone, Textile, Wall and Wood for a total of 142 different textures.

- **Rotation & Flip -**
  The image has a 50% chance to be horizontally flipped, and n rotation is randomly picked between 0 and 3. Then the image is rotated n rotation times by 90ř.

- **Saturation -**
  Contrarily to dull colours, it is quite rare to obtain over saturated colours with digital cameras nowadays. Thus, the saturation of each image is randomly modified from -50%(dull colours) to +20% (saturated colours). Consequently, the newly generated images are more frequently under-saturated.

- **Contrast -**
  The image is randomly contrasted with a coefficient randomly selected from -50% to +50%

- **Brightness -**
  Finally, the brightness is also randomly modified with a range of -70% to +20% for similar reasons to the saturation random modification process.

  Other than this there are also many more data augmentation techniques to increase the preprocessing data. Data augmentation will help to build the Deep Learning Model more robust and avoid overfitting.

  To summarize, for each CAD Model, 38 screenshots are taken automatically from different viewpoints all around the models. Then each screenshot is processed by the data augmentation pipeline. And thus leading, for each CAD Model, to: $no.of screenshots * (augmentation_factor + 1) = 38 * (30 + 1) = 1178$ images (1 is added to the augmentation factor because the original screenshot is also kept unmodified in the process).

2.4.2 Deep Learning - Model Selection and Tuning

Building deep learning models might be a tedious and complex process. Selection of deep learning model consists of -
selection of no. of hidden layers - This is important parameter to select as increase in no. of hidden layers increases the computation cost and model tends to overfitting which will effect on the accuracy on test data.
No. of neuron in each layer - the number of hidden neurons does affect the model performance. When a neural network has too few hidden neurons (< 16), it does not have the capacity to learn enough of the underlying patterns effectively. When the neural network has $>= 16$ neurons, the neural network start to do better. At

an increasing number of hidden neurons (>= 128), the number of hidden neurons does not help too much for this problem.[7]
Activation functions - There are different activation functions used in deep learning like sigmoid function, tanh, ReLU(Rectified Linear Unit), Leaky ReLU. ReLU gives better result as compared to others for most of the applications [7].

Optimization of all the above parameters is quite tedious for all training data as it will be computationally costly. It's better to run the model on a small dataset and after optimizing these parameters run the model on full datasets.

Model Tuning

To construct a deep learning model, a myriad of parameters (i.e. weights of the ML or DL model which are fit during the training) and hyperparameters (i.e. higher-level parameters which are determined by the data scientist before the training process) needs to be determined, tuned and set.

The table below presents a few noteworthy hyperparameters which are necessary to determine and correctly set in a deep learning model. They have been sorted in two categories: the ones on left relate to the structure of a deep learning model and those on the right influence the behaviour of the model. Both of these categories are essential to obtain accurate and robust results.

Most of the time, data scientists make assumptions on certain parameters (e.g. the type of network used) or establish a range of values that a parameter can take. After that, the model will be trained and cross-validated, across the possibilities considered, searching for the best combination of hyper-parameters. Some heuristics exist, in the literature, to facilitate or enable this process [5].

Architecture hyperparameters	**Global hyperparameters**
Type of Input	Loss Function
Network Type(e.g. CNN, RNN, etc.)	Learning Rate
No. of hidden layers	Type of regularization
No. of neurons in each layer	Weight optimization algorithm
Type of layer	learning rate decay rate
How to initialize the network	Fitness matrix used
Activation Function	Conditions for early stopping
Dropout rate	No. of Epochs

Table 2.1: Some of the key hyperparameters that need to be determined and set during the conception of a deep learning model[5]

2.4.3 Training and Testing Set

It is good practice to split the dataset into two parts: the training set and testing set. It is necessary to determine how the proposed dataset will be split into two parts. Randomizing the data repartition is essential to limit the possibilities of an inner bias, inside the training set, which will be later learned by the ML or DL model. With a bigger dataset, up to 80% of the data could have been included in the training set. Reducing this ratio down to 50% will help to detect overfitting when the amount of available data is limited. Overfitting can be considered as learning the noise inside the data, which highly reduce the effectiveness and accuracy of the model.

K-Fold cross-validation is used to train the model in an efficient way such that it avoids the bias from the data. "Cross-validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. In typical cross-validation, the training and validation sets must cross-over in successive rounds such that each data point has a chance of being validated against. The basic form of cross-validation is K-Fold Cross-validation."

One of the problems with K-Fold Cross-validation, in a classification problem, is that some of the folds may present a very different class distribution from the original dataset. At the extreme, some folds may present only one class thus introducing bias in the metrics.

Stratified K-Fold helps to address this issue by reproducing the original class distribution in the K-Folds randomly generated. Therefore, it is good to apply a Stratified K-Fold Cross validation process during the training process.

Once the training on the network is completed, the accuracy of the model is calculated by running the model on the test data. Table 8 presents the results obtained with each DL model with and without transfer learning being used.

Model Name	**Accuracy(Training%)**	**Accuracy(Test%)**
Random guess	3.3	3.3
LeNet-5	14.16	7.4
AlexNet	63.96	34.67
AlexNet(Transfer Learning)	58.73	52.50
GoogLeNet	90.17	51.76
GoogLeNet(Transfer Learning)	84.35	82.81

Table 2.2: Summary of the results obtained during the DICE case-study [5]

As we can see that GoogLeNet gives more accurate result as compared to other model.

2.5 Machining feature recognition based on 3D CNN

Automated machining feature recognition, a sub-discipline of solid modelling, has been an active area for last three decades and is a critical component in digital manufacturing information from CAD models. In Zhang et al. [12] paper deep 3D CNN is used to learn machining features from CAD Models of mechanical parts is presented. FeatureNet learns the distribution of complex manufacturing feature shapes across a large 3D model dataset and discovers distinguishing features that help in recognition process automatically. To train FeatureNet, a large-scale mechanical part dataset od 3D CAD Models with labelled machining features from the low-level geometric data such as voxels with very high accuracy.

In Zhang et al. [12] proposed deep learning approach that can handle a large data set of complex feature recognition. The proposed deep learning approach uses multiple convolution layers to learn a robust and accurate model for recognizing 3D machining features. The performance of any machine learning system is highly dependent on the quality of dataset used. Therefore, we generate a large dataset of different classes of features in different sizes and orientation. The proposed data-centred deep convolution neural network is trainable, i.e. using a different dataset, it would be possible to re-train the network to recognize a different set of features.

Zhang et al. [12] created feature dataset created consists of 24 unique and commonly occurring machining features. For each feature, 6000 different samples were generated. Randomly sampled sizes and all six orientations were used to generate feature dataset. The variability in the dataset makes the trained model more robust and accurate. A deep 3D Convolution Neural Network(3D-CNN) was trained to recognize machining features using the generated dataset. The proposed 3D-CNN, termed FetureNet, consists of eight layers - an input layer, four convolution layers, a pooling layer, and a final soft max classification layer. TensorFlow library was used to implement and train the CNN network. About 34 Million parameters of the network were learned using 144,000 3D models in the machining feature dataset. The feature recognition accuracy obtained was 96.7 % .

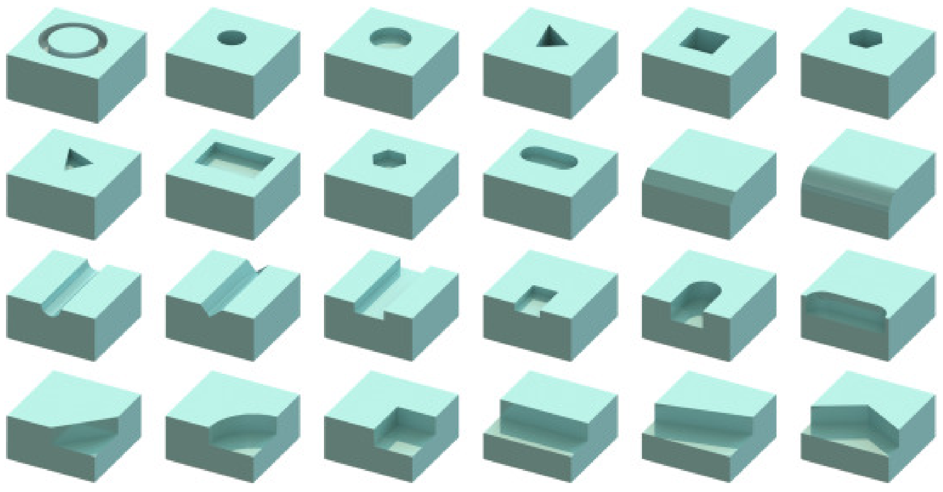


Figure 2.7: A set of 24 machining features used for classification [12]

This network only recognizes non-intersecting features.
Data creation of CAD models to train the model is given in Figure 2.8 below-

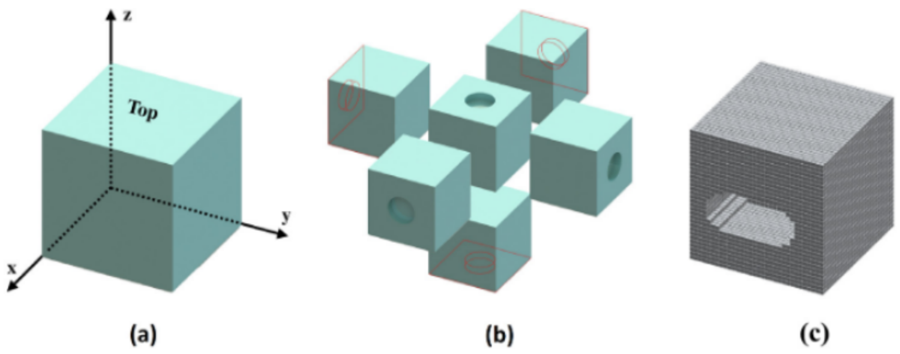


Figure 2.8: Data generation pipeline [12]

Voxel resolution	Training time(min)	Classification time(ms)	Classification Accuracy(%)
$16 \times 16 \times 16$	7.5	1.27	83.2
$32 \times 32 \times 32$	54	5.05	93.7
$64 \times 64 \times 64$	390	33.04	97.4

Table 2.3: Effect of voxel resolution on computation time and classification accuracy [12]

Above table shows the effect of voxel resolution on the time and accuracy. For this study, Zhang et al. [12] selected a subset of dataset with 7 machining features and 2400 examples for each feature and model is trained on this dataset with

same hyperparameters. A typical CAD component usually has multiple machining features. Often, some of those features are overlapping. But the FeatureNet was trained to identify component with single a feature only. Again this network will not recognize feature in feature.
Ghadai et al. [6] used deep learning to learn different DFM rules associated with design and manufacturing. [6] paper was able to learn the complex DFM rules for drilling, which include not only the depth-to-diameter ratio of the holes but also their position and type (through hole vs blind). The framework can be extended to learn manufacturable features for a variety of manufacturing processes such as milling, turning, etc. By training multiple networks for specific manufacturing processes, which can be concurrently used to classify the same design concerning their manufacturability using different processes. Thus, an interactive decision-support system for DFM can be integrated with current CAD systems, which can provide real-time manufacturability analysis while the component is being designed. This would decrease the design time, leading to significant cost-savings.

Figure 2.9 shows a framework to decide whether the designed component will be manufactured or not. The CAD model is converted to a voxel representation and is input to a 3D-CNN for manufacturability classification. The 3D-CNN output is analyzed using 3D-GradCAM to provide manufacturability feedback.

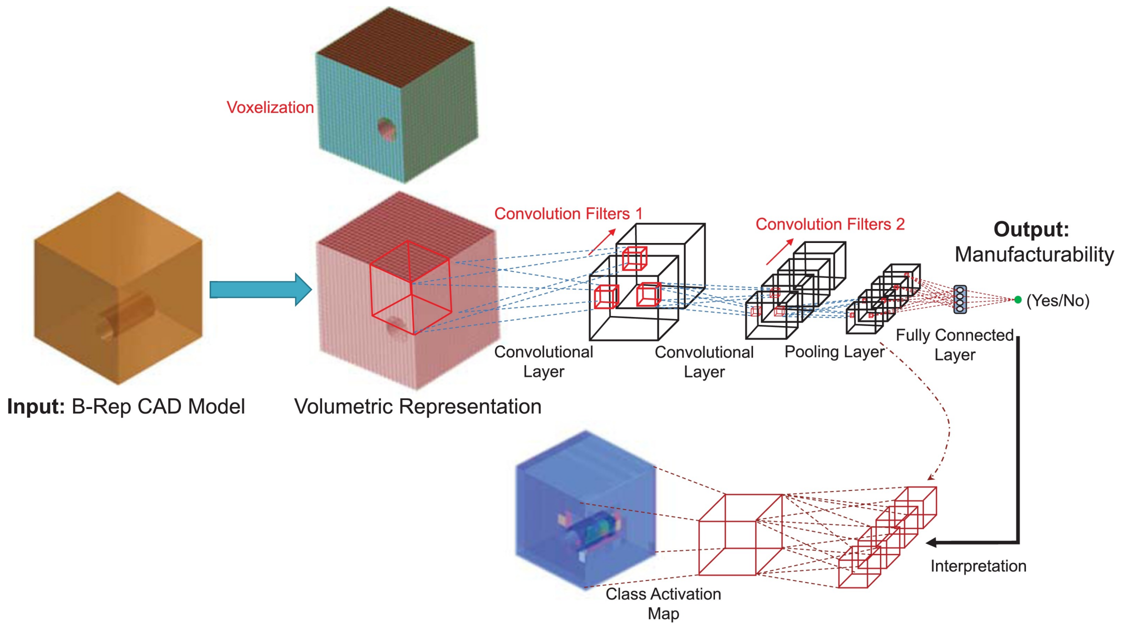


Figure 2.9: Framework for deep-learning based design for manufacturability [6]

The input to the 3D-CNN is a voxelized CAD model. The input volumetric data is first padded with zeros before convolution is performed. Zero padding is necessary in this case to ensure that the information about the boundary of the CAD model is not lost while performing the convolution. The convolution layer with RELU activation is followed by a batch normalization layer and a Max.

Pooling layer. The same sequence of Convolution, batch normalization, and Max. Pooling is again used. A fully connected layer is used before the final output layer with sigmoid activation. The model parameters θ comprised of weights W and biases, b are optimized by error back-propagation with binary cross-entropy as the loss function using the ADADELTA optimizer [6]. The loss function l to be minimized is:

$$l = -ylog(\hat{y}) - (1 - y)log(1 - \hat{y})$$

Where $y \in [0, 1]$ is the true class label and $\hat{y} \in [0, 1]$ is the class prediction. Quantitative performance on test data sets are given below -

Test Data	True Positive	True Negative	False Positive	False Negative	Accuracy
675 models	391	90	17	176	0.7136

Table 2.4: Effect of voxel resolution on computation time and classification accuracy [6]

2.6 Defeaturing of CAD Models

Numerical simulations play more and more important role in product development cycles and are increasingly complex, realistic and varied. CAD models must be adapted to each simulation case to ensure the quality and reliability of the results. The defeaturing is one of the key steps for preparing a digital model to a simulation. It requires great skill and deep expertise to foresee which features have to be preserved and which features can be simplified. This expertise is often not well developed and strongly depends on the simulation context. In [4] paper they have proposed an approach that uses machine learning techniques to identify rules driving the defeaturing step. The expertise knowledge is supposed to be embedded in a set of configurations that form the basis to develop the processes and find the rules. For this, [4] propose a method to define the appropriate data models used as inputs and outputs of the learning techniques.

In the field of numerical simulation, the adaptation and idealization processes of CAD models to prepare simulation models can be seen as a completion of complex tasks involving high-level expertise and many operations whose parameterization relies on a deep knowledge Not often clearly formalized. To address this problem, machine learning techniques can be a good mean to find rules that drive the CAD models preparation Processes. Moreover, those techniques can be very helpful to capitalize the knowledge embedded in a set Of adaption scenarios. Depending on the objective of the targeted simulation (structural, dynamic/fluid/heat transfer simulations, assembly/dis assembly procedure evaluations), as well as on the type of method adopted for solving it (Finite Differences, Finite Elements Analysis and so on), there exists a large amount of possible treatments to prepare the simulation model from an initial CAD model (feature removal, part or sub-product removal, shape simplification, size reduction, meshing, meshing adaptation). Figure 2.10 shows multiple representations a CAD model may have depending on the targeted simulations. These treatments can be applied in different orders and with different tools. Treatment sequences strongly affect the processing time as well as the result accuracy of the simulation solutions.

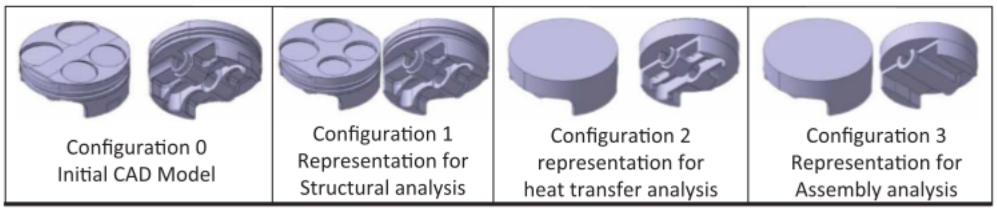


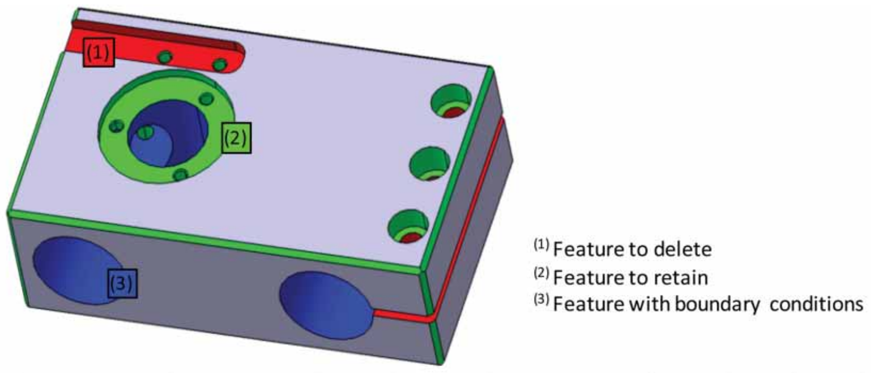
Figure 2.10: Multiple representations of a CAD model adapted to different simulation objectives [4]

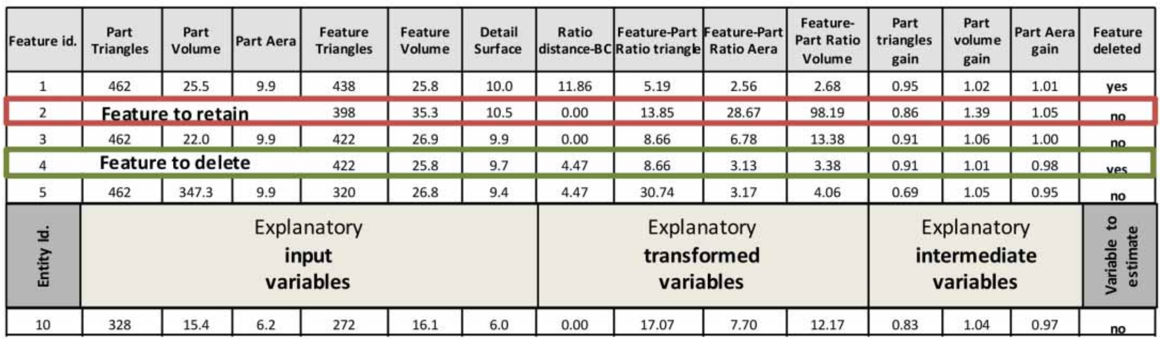
Removing unnecessary features according to a given simulation objective. Indeed, these suppression's are supposed not to affect the simulation results, and they can strongly speed up the overall simulation process. Today, these preparation processes are not well developed. Currently, to select the best treatment, expert choices are based on indicators such as the targeted accuracy, cost or preparation time. These indicators must take into account a large num- ber of data relative to the simulation objective, CAD Model characteristics and practical industrial constraints. Thus, the evaluation criteria are difficult to quantify and generalize. Particularly, when selecting features (e.g. rounds, chamfers, holes, bumps, pockets, free-form features) to be deleted or preserved, the choice is often empirical and leads, by precaution, to be more precise than necessary (i.e. some features are preserved when they could be removed).

Danglade et al. [4] used machine learning to capitalize the expert knowledge and to provide decision support during the defeaturing steps. They will use these tools to estimate numeric variables (e.g. duration and cost of preparation or simulation) or qualitative variables (e.g. features classification depending on their size “small / medium / large”, relative position to the boundary conditions “feature=BC, feature near BC, feature away from the BC,...”) for which no rules are formalized. Then, from these variables, machine learning tools provide rules to propose candidates for defeaturing.

The first step in machine learning is to create a database that contains a set of collected information. The initial database must include a significant number of already known configurations and scenarios. The two data sets the input one (CAD model characteristics before simplification, designers' needs and industrial constraints), and the output one (CAD model characteristics after simplification, process settings and performance indicators) are then identified. The CAD model characteristics correspond to information like the type of CAD model (e.g. component or assembly), the format (e.g. CATIA native, STEP, IGES, tessellated model), the material, the component family (e.g. longitudinal, solid, tubular, thin) and the dimensional quantities (e.g. size, surface area, volume, number of triangles, number of faces).

The learning database is a spreadsheet shown on Figure **??** whose rows contain targeted learning entities. The columns (except the last one) contain the explanatory input and intermediate variables, and the last one contains the output variable to estimate. The main objective of this study is to identify these fields and to ensure their completeness. We have to begin choosing the aims (variable to estimate in the last column) and the targets of learning (rows). Then, we will select, transform and repair data from the initial database to create a database for learning.





Feature id.	Part Triangles	Part Volume	Part Aera	Feature Triangles	Feature Volume	Detail Surface	Ratio distance-BC	Feature-Part Ratio triangle	Feature-Part Ratio Aera	Feature-Part Ratio Volume	Part triangles gain	Part volume gain	Part Aera gain	Feature deleted
1	462	25.5	9.9	438	25.8	10.0	11.86	5.19	2.56	2.68	0.95	1.02	1.01	yes
2	Feature to retain			398	35.3	10.5	0.00	13.85	28.67	98.19	0.86	1.39	1.05	no
3	462	22.0	9.9	422	26.9	9.9	0.00	8.66	6.78	13.38	0.91	1.06	1.00	no
4	Feature to delete			422	25.8	9.7	4.47	8.66	3.13	3.38	0.91	1.01	0.98	yes
5	462	347.3	9.9	320	26.8	9.4	4.47	30.74	3.17	4.06	0.69	1.05	0.95	no
Entity Id.	Explanatory input variables						Explanatory transformed variables				Explanatory intermediate variables			Variable to estimate
10	328	15.4	6.2	272	16.1	6.0	0.00	17.07	7.70	12.17	0.83	1.04	0.97	no

Figure 2.11: Learning database spreadsheet for defeaturing objective [4]

The result obtained by using various machine learning algorithm is listed in table below -

Cross Validation tests	**Training Accuracy**	**Test Accuracy**
Neural Networks	99.2	91.2
Support Vector Machine	66.7	61.4
Decision Tree	93.5	87.7

Table 2.5: Result on training and test data using different machine learning method [4]

From the above table it is clear that the models giving the best results are neural networks. This study is limited to analysis stress and heat transfer simulations for a single parts, it may be extended to other targets and other types of CAD models.

2.7 Overall Observations

It has been observed that deep learning algorithms and routines may present many desirable advantages and assets to solve challenges inherent to the industry of the future.

With the increasing amount of available data, computing power and network speed for a decreasing cost, the manufacturing industry is facing an unprecedented amount of data to process, understand and exploit. We need to do preprocessing of 3D CAD models to feed the network. ML and DL can be used in CAD domain for feature extraction of CAD models; DL models gives higher accuracy to recognize the different features. Also, it can be used to classify CAD models into different classes. Topology optimization of component can be done by using Deep Learning.

Deep Learning can be effectively used to takes in one or more images of an object instance from arbitrary viewpoints and outputs a reconstruction of the object in the form of a 3D occupancy grid. This will help to build a 3D model from single or multiview.

Dekhtiar et al. [5] explained how deep learning models appear to be highly efficient to address problematics of the manufacturing industry. To pursue in that direction, possibilities offered by some deep learning models will be explored in future works.

Chapter 3

Conclusions

3.1 Overview

This report studies various application of Machine Learning applications related to CAD domain. There are more challenges in object detection and preprocessing of the CAD model data. The recent advancement in Deep Learning can help to build robust network that can train on unbiased dataset to achieve higher accuracy. There is a lot of research work is going on to develop higher level model that will increase the accuracy of network.

3.2 Conclusion and Scope of Work

Machine learning is one of the most highly actively researched and sought after topic in recent times. Deep learning is giving much better result as compared to other traditional method.

There are areas in CAD domain where we can use Machine Learning to learn from the existing data. Currently there are more work is going on in the field of object classification, Feature extraction, 3D model Reconstruction from 2D images, Defeaturing of CAD models for simulation. There is more scope to improve accuracy of model for each case by optimizing the architecture of neural network. To summarize, deep learning algorithms and routines may present many desirable advantages and assets to solve challenges inherent to the industry of the future or industry 4.0.

Bibliography

[1] Christopher M Bishop. *Pattern recognition and machine learning.* springer, 2006.

[2] Medium Corporation. *Popular Machine Learning Algorithms*, 2019 (accessed April 1, 2019). `https://medium.com/technology-nineleaps/popular-machine-learning-algorithms-a574e3835ebb`.

[3] Akihiro Daito, Tsutomu Ohashi, Takashi Murosaki, Masahiro Nishiwaki, and Ichiro Kimura. Predictive diagnosis of machine failure with deep learning method. In *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 949–954. IEEE, 2017.

[4] Florence Danglade, Jean-Philippe Pernot, and Philippe Véron. On the use of machine learning to defeature cad models for simulation. *Computer-Aided Design and Applications*, 11(3):358–368, 2014.

[5] Jonathan Dekhtiar, Alexandre Durupt, Matthieu Bricogne, Benoit Eynard, Harvey Rowson, and Dimitris Kiritsis. Deep learning for big data applications in cad and plm–research review, opportunities and case study. *Computers in Industry*, 100:227–243, 2018.

[6] Sambit Ghadai, Aditya Balu, Soumik Sarkar, and Adarsh Krishnamurthy. Learning localized features in 3d cad models for manufacturability analysis of drilled holes. *Computer Aided Geometric Design*, 62:263–275, 2018.

[7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016.

[8] Hongbin Jia, Yi Lu Murphey, Jinajun Shi, and Tzyy-Shuh Chang. An intelligent real-time vision system for surface defect detection. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 239–242. IEEE, 2004.

[9] Pavan Kumar Kankar, Satish C Sharma, and Suraj Prakash Harsha. Fault diagnosis of ball bearings using machine learning methods. *Expert Systems with applications*, 38(3):1876–1886, 2011.

[10] mathworks. *Classification of ML Algorithms*, 2019 (accessed March 9, 2019). https://in.mathworks.com/help/stats/machine-learning-in-matlab.html.

[11] Andrew NG. *Classification of ML Algorithms*, 2019 (accessed February 2, 2019). http://cs231n.github.io/neural-networks-1/.

[12] Zhibo Zhang, Prakhar Jaiswal, and Rahul Rai. Featurenet: Machining feature recognition based on 3d convolution neural network. *Computer-Aided Design*, 101:12–22, 2018.