AI in Manufacturing CAD Models

RISHABH KUMAR SINGH

26 February 2022

Abstract

In this report, I have proposed the idea of How we can predict the price of the manufacturing of the CAD models, Manufacturing CAD models sometimes seems to be a very typical task when it comes to manufacturing some complex structures like a ball bearing, nuts, bolts and many and deciding the manufacturing price of it is also a very complex task and require an industry expert and getting an industry expert always is not possible everywhere, so if there is a web app in which we put a cad model or a point cloud model and get an estimated price of the CAD or Point Cloud Model would be nice and helpful.

1. Problem Statement

The problem statement is applied to the manufacturing of the CAD models in which our task is to get an estimated manufacturing price of a particular CAD model, without any intervention from an industry expert.

2. Market/Customer/Business needs Assessment

When taking a look at the market there is no automated tool that can estimate the manufacturing price of the CAD model priorly, we need someone who can estimate its manufacturing price by taking a look at

various features like shape, size, and others. Having a tool that can predict the price will be helpful for all the manufacturers and escalate their business

3. Target Specification

The proposed service will be given in the form of a web app in which the manufacturer needs to submit the CAD Model file or a Point Cloud Image, and the web app will show him the resulted manufacturing price of the CAD model, the price is not the actual one but mostly resemble the price provided by the industry expert.

4. External Search

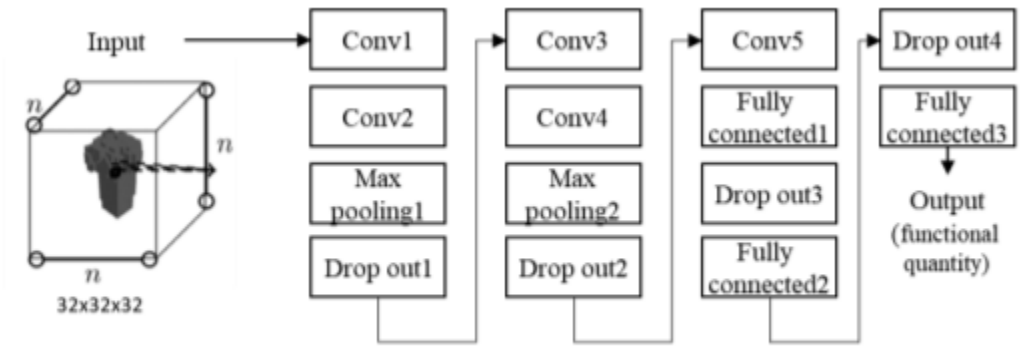
These are some research papers which I have taken a reference to come up with the above idea

- https://www.researchgate.net/publication/338326681_Manufacturing_cost_estimation_based_on_a_deep-learning_method
- https://arxiv.org/ftp/arxiv/papers/2010/2010.14824.pdf

The dataset which is going to be used in this project is a set of voxel and cad models which is available in this dataset repository. The dataset consists of 4000+ CAD models which are going to be used in training a 3D CNN.

A 3D CNN is being trained on the 3D CAD Models, the 3D CNN model is mostly build-up by using VGG16 as a backbone. The VGG16 is being chosen due to having less number of layers which help us to train our model from scratch without performing any kind of training.

Below is the structural diagram of the basic 3D CNN in which we are feeding a 3D Mesh/Voxel/CAD/Point cloud model.



(b) 3D CNN arthitecture of Dering and Tucker. (2017)

Cite: https://arxiv.org/ftp/arxiv/papers/2010/2010.14824.pdf

After classifying a particular 3D CAD model now we need to check the category onto which the current model, belongs, and then we can pass its properties to get its estimated price in a regressor model, the regressor model can be a basic regression model like RandomForestRegressor or an Xgboost regressor now this price is an estimated price which is given by the regressor model which is roughly correct or near to the actual one which is given by the industry expert.

5. Concept Generation

The first step is to preprocess a 3D CAD model the step involves the mesh file conversion and Voxel and point cloud transformation. The mesh file conversion will help us to get a Point cloud format which we can further send to the pointnet++ for some classification, point net is just to test a particular network, as I also take 3D CNN for classification. After that, the features of the 3D CNN model are then fed into the regressor model, and then a web app will be created to add all those components.

6. Basic Code Design

Below is the code block for the Pointnet which is majorly used to classify the 3D CAD models. Pointnet is the combination of 3D convolution layers and dense layers.

```
def tnet(inputs, num_features):

    # Initalise bias as the indentity matrix
    bias = keras.initializers.Constant(np.eye(num_features).flatten())
    reg = OrthogonalRegularizer(num_features)

    x = conv_bn(inputs, 32)
    x = conv_bn(x, 64)
    x = conv_bn(x, 512)
    x = layers.GlobalMaxPooling1D()(x)
    x = dense_bn(x, 256)
    x = dense_bn(x, 128)
    x = layers.Dense(
        num_features * num_features,
        kernel_initializer="zeros",
        bias_initializer=bias,
        activity_regularizer=reg,
    )(x)
    feat_T = layers.Reshape((num_features, num_features))(x)
    # Apply affine transformation to input features
    return layers.Dot(axes=(2, 1))([inputs, feat_T])
```

```
inputs = keras.Input(shape=(NUM_POINTS, 3))

x = tnet(inputs, 3)
x = conv_bn(x, 32)
x = conv_bn(x, 32)
x = tnet(x, 32)
x = conv_bn(x, 32)
x = conv_bn(x, 64)
x = conv_bn(x, 512)
x = layers.GlobalMaxPooling1D()(x)
x = dense_bn(x, 256)
x = layers.Dropout(0.3)(x)
x = dense_bn(x, 128)
x = layers.Dropout(0.3)(x)

outputs = layers.Dense(NUM_CLASSES, activation="softmax")(x)

model = keras.Model(inputs=inputs, outputs=outputs, name="pointnet")
model.summary()
```

6. Product details - How does it work?

An interactive web app will be created at the end with all the functionalities attached in which the user can enter the mesh file and the price of the CAD model will be displayed.