Internship Project
Report

Sheet Metal Part Identification using Machine Learning

Submitted by

Priyadharsshni
Sampad Kumar Kar
Shankar Ram Vasudevan

Under the guidance of

Sanjiv Doraiswamy
Trumpf Metamation

Trumpf Metamation
SIPCOT IT Park, Siruseri, Chennai

Internship 2022-23

Trumpf Metamation

Certificate

Name of the guide
(Project Guide)

Date:

Abstract

Recent trends in the manufacturing industry have shown a growing interest in automation and Industry 4.0. This is driven by the need to improve productivity, reduce costs, and increase the overall quality of products. One of the key areas where automation can have a significant impact is in the classification and handling of sheet metal parts.

One approach that has been proposed in the literature ([1], [2]) is the use of computer vision and machine learning techniques to automatically identify and classify sheet metal parts. These techniques can be used to analyze images of the parts and extract relevant features, such as shape, size, and surface texture. The extracted features can then be used as input to a machine learning model, which can accurately classify the parts.

This project aims to develop a pipeline for classifying sheet metal parts and return the corresponding CAD file.

In addition to improving accuracy, automating the classification of sheet metal parts can save time and money. By reducing the need for manual inspection and sorting, the system can help to increase the overall efficiency of the manufacturing process.

Contents

Chapter 1

Objective

The main objective of this project is to create a pipeline that can accurately classify a given image as a sheet metal part or not, and output the corresponding CAD file if it is. The problem can be further broken down into three steps

1. Collect relevant labeled data (images) and apply pre-processing techniques to prepare it for training machine learning models.
2. Choose the appropriate architecture for the model and optimize the hyperparameters to improve accuracy.
3. Deploy the model and make it accessible to the customers.

In addition to this, the model must be built in a modular way to keep up to date with the rapid developments in the field of Computer Vision algorithms. In the following chapters, we explain in detail the different models we have implemented, their downfalls, and the potential ways to address them in the future.

Chapter 2

Introduction

2.1 Image Processing

Image processing is the method of manipulating and analyzing digital images using mathematical algorithms. The process can be divided into two main categories: analog image processing and digital image processing. Analog image processing involves performing operations on an image as a continuous signal, while digital image processing deals with the manipulation and analysis of digital images, which are discrete in nature. Image processing techniques are used for a wide range of applications, such as object recognition, medical imaging, satellite imagery, and video compression. Some popular techniques used in image processing include image enhancement, contrast correction, and perspective warping.

2.2 Fundamentals of Machine Learning

2.2.1 Classical ML

Classical Machine Learning refers to the traditional algorithms that were prevalent before the advent of Deep Learning. It includes techniques such as linear regression, logistic regression, Support Vector Machines, and Decision Trees. These models require significantly lesser computation resources compared to Deep Learning models and are also relatively easy to interpret and understand, making them useful in fields such as medical diagnostics and financial forecasting.

2.2.2 Deep Learning

Deep learning is a subset of machine learning that uses algorithms inspired by the structure and function of the brain's neural networks. These models are called artificial neural networks (ANNs) and are made up of layers of interconnected nodes, called artificial neurons. The ANNs are trained on large amounts of data by adjusting the connections and weights of the neurons, allowing the model to learn complex patterns and make predictions.

Convolutional Neural Networks (CNNs) are a special type of Neural Network first introduced by Yann LeCun in [3]. They are particularly useful for tasks such as image recognition and classification. CNNs achieve this by looking for local patterns in images which are then aggregated to find global patterns.

2.3 Computer Vision

Computer Vision is a field of study that aims to enable computers to process visual data such as images and videos and capture complex patterns, relying on techniques mentioned above such as Image Processing, Classical ML, and Deep Learning.

One of the earliest applications of Computer Vision and Machine Learning was in the manufacturing industry. According to a paper [4] published in the Mechanic Automation and Control Engineering in 2010, automated inspection systems using machine vision technology were developed to detect surface defects in plastic.

Chapter 3

Work Done

3.1 Preparation and Background Research

3.1.1 Image Processing

Since real-world images require extensive pre-processing before being ready for model training, we completed an Image Processing course (by Prof. Kavita Sutar, CMI) that covered various fundamental concepts such as homography, camera calibration, contrast correction, edge detection, and object detection. The code for the assignments demonstrating our understanding of basic image processing techniques is available in the GitHub repository provided.

3.1.2 Machine Learning and Deep Learning

The basic concepts of machine learning were reviewed through a course, specifically the Python for Data Science and Machine Learning Bootcamp from Udemy. This course covered various supervised and unsupervised algorithms, such as logistic regression, support vector machine, principle component analysis, and many others. To further understand the workings of neural networks and deep learning, another course was taken, the Intro to Deep Learning with PyTorch from Udacity. This course explained the workings of neural networks, the role of weights, gradient descent, and backpropagation. Additionally, it covered convolutional neural networks and recurrent neural networks.

3.1.3 Advanced Topics in Deep Learning

Residual Networks

Residual Networks or ResNets ([5]) is a state-of-the-art CNN architecture used for image classification tasks. Trained on over 1.3 million images consisting of 1000 classes, they use "skip connections" which allows the training procedure to converge faster. To test the performance of ResNet, we built an Emotion Recognition software that uses facial features to accurately predict the emotion. The code is available in the GitHub repository provided.

Transfer Learning

Transfer learning is the process of using knowledge from one task to improve the performance of a related but different task. This technique addresses the following two issues :

- Models trained on small datasets tend to overfit and have poor accuracy on unseen data
- Training models on large datasets is computationally infeasible

Transfer learning allows us to harness the power of large-scale pre-trained models by fine-tuning them on a much smaller dataset to improve accuracy.

Generative Adversarial Networks

Generative Adversarial Network (GAN) is a generative model that consists of a Generator and a Discriminator [6]. GANs are widely used in state-of-the-art models to produce text and images that are realistic and reflective of the dataset they are trained on. The Generator and the Discriminator play a two-player game with the objective of the Discriminator being the accurate segregation of real and fake images and the Generator fooling the Discriminator into thinking the synthetic images are real.

After training the model, the Generator can be used to produce realistic images which can be used for training. We evaluate the performance of the DCGAN [7] by training it on the MNIST dataset. The code is available in the GitHub repository provided.

3.2 Data Collection

Since machine learning algorithms are relatively new in the sheet metal industry, there do not exist good datasets which have labeled, pre-processed

images. Due to this reason, we resorted to crawling the web using the Bing image search API to obtain sheet metal part images. These were then manually sorted to weed out poor-quality or watermarked images. Finally, some common data augmentation techniques were applied to produce a dataset that can be used for subsequent model training. The data and the code used to obtain it are available on the GitHub repository provided.

3.3 Model Building

3.3.1 Phase I: Classification of images as SMP/non-SMP

The first phase of the project is to build a model which can accurately classify an input image as a Sheet Metal Part (SMP) or non-Sheet Metal Part (non-SMP). This is a one-class classification problem and we faced the following issues while building a model for the same.

Difficulties:

- Overfitting: Since there is only one class of images (SMP) to learn from, the model might overfit on the training data and have poor generalization error.
- Limited Feature Representation: If the model is not complex enough to capture the features of the target class, the overall accuracy would be low.

To overcome these issues, we decided to use the transfer learning paradigm outlined earlier, leveraging the ResNet18 model pre-trained on the ImageNet dataset. The diversity of the ImageNet dataset and the deep architecture of the ResNet18 model gives us great results, as outlined in the next chapter. The code is available in the GitHub repository provided.

3.3.2 Phase II: Matching SMP with corresponding CAD files

The second phase of the project is to develop an algorithm that given an SMP image returns the corresponding CAD file.

Difficulties:

- Isolating the SMP from the background can be a difficult task due to the presence of other metal parts in the background, and the angle and lighting under which the image is captured.
- Edge detection algorithms are not robust enough to filter out the noise that is captured in the image. Hence, the matching algorithm might not return the corresponding CAD file.

We address the first problem by using the SOTA Image Segmentation model U^2-net [8], which removes the background while retaining the Sheet Metal Part. Some examples of the background removed images are provided in the repository for context and evaluation.

The second challenge mentioned above requires further work. We will outline the methods we have tried and possible future directions.

We tried tackling this obstacle by implementing the following edge detection and matching algorithms in the pipeline.

1. Gaussian Blurring + Canny Edge Detection: Canny Edge Detection captures a lot of the noise in the image. Gaussian Blurring deals with it to a small extent, but the performance is still poor. Trying out different parameters for blurring and other standard edge detection algorithms might reveal some insights.
2. SIFT detector: Since the SMP in the image could be rotated or of a different scale, we need a robust enough matching algorithm that can capture and compare local features. To this extent, we experiment with the SIFT detector detailed here [9].
3. We use VGG-16 pre-trained on ImageNet as a feature extractor and find the closest match in the feature space wrt L2 norm.
4. We use pre-trained DexiNed [10], SOTA for edge detection to extract the edges in the image.

Due to the lack of relevant data (real SMP images and corresponding CAD drawings), all of the techniques perform poorly on the image-matching task in the real world. Under controlled settings, where the sheet metal part is placed on a solid-colored background, the algorithms perform much better. Given additional paired image data, one possible approach to extend the application of this pipeline to the typical industrial setting (the background has some noise, and other sheet metal parts are present in the image) is to fine-tune a pre-trained generative model that produces the sketch of a given image on this dataset.

Chapter 4

Results

For Phase I, ResNet18 pre-trained on ImageNet-1k, was retrained on our dataset with initial layers frozen and the last FC layer initialized with weights from the standard normal distribution. The model was trained for 30 epochs and yielded a test accuracy of 96.5%.

During real-time inference, the model correctly predicted 16 out of 20 images. We attribute this drop in accuracy to a lack of domain knowledge when labeling the dataset. Retraining the model on a better dataset will fix this issue. **Figure 4.1** shows the prediction of the classifier on some test images.

For Phase II, the state of the art background removal tool (built on top of U^2-net [8]) produced excellent results for images with complex backgrounds and poor lighting conditions. To demonstrate the prowess of this tool, we test this out on raw images of Sheet Metal Parts as shown in **Figure 4.2**. Even on images with uncontrolled background, we managed to separate the sheet metal parts from their background, as shown in **Figure 4.3**.

Post background removal, the state of the art edge detector (DexiNed) was used to detect edges in the images with high accuracy. However, there were times when the model struggled to detect edges correctly. This was mostly observed when the sheet metal part was hollow (like the second sheet metal part in **Figure 4.2**). This is because the background leaks in noise (like the wooden background), which confuses the edge detector. **Figure 4.4** shows the edge detected sheet metal parts corresponding to the same set of images and how it compares against the original cad drawings as shown in **Figure 4.5**)

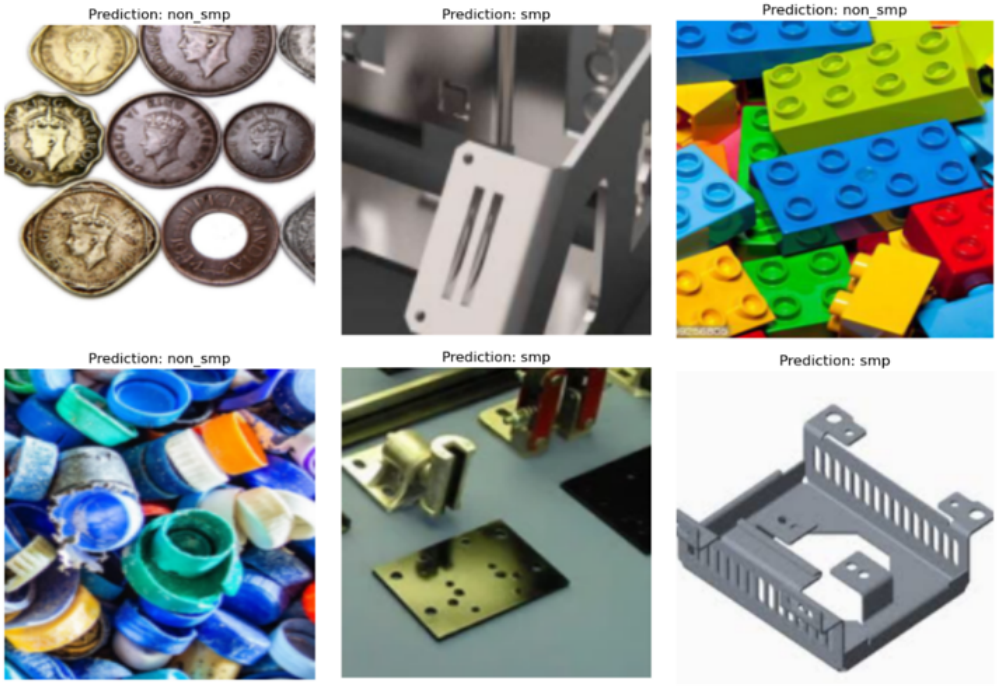

Figure 4.1: Prediction of classifier on some test examples.

Figure 4.2: Raw images of sheet metal parts.

Figure 4.3: Background removed images.

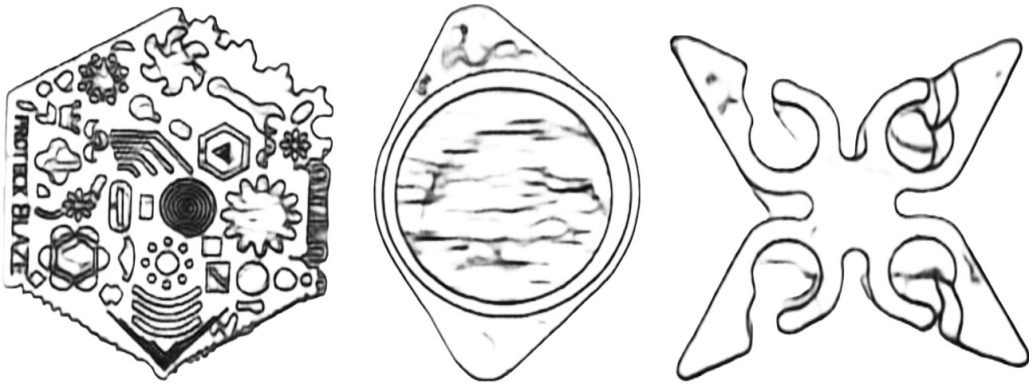

Figure 4.4: Edge detected images.

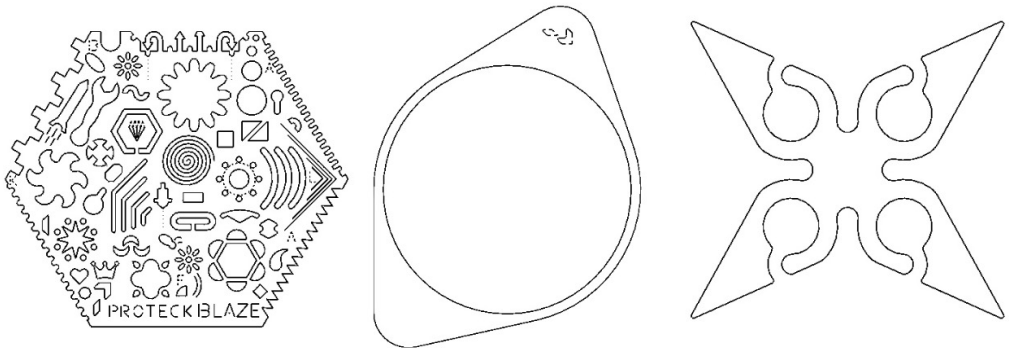

Figure 4.5: CAD drawings corresponding to the raw sheet metal parts.

Chapter 5

Future Work

We plan to deploy a pipeline for identifying sheet metal parts in images as a web application. The pipeline includes two main steps:

- classifying the image as a sheet metal part or not,
- labeling the sheet metal part as one of the 10 pre-labeled classes of sheet metal part types.

The final output of the pipeline will be a stored CAD file corresponding to the identified sheet metal part type. We plan to deploy this in the form of an interactive Web-App, which can potentially be scaled up for commercial usage as well. This can be achieved in the following 4 steps:

5.1 Model Export

We can export the sheet metal parts vs non-sheet metal parts classifier and the sheet metal type classifiers as `.h5` files, which can directly be used by our Web App without training.

5.2 Web Framework

- **Front-end:** We can use `HTML`, `CSS` (with Bootstrap) in the front-end.
- **Back-end:** We can use `Flask` as the web framework for the Web-App and `Flask SQLALchemy` is used to interact with the database agent.

5.3 Functionality

- The app will allow users to upload an image, which will be passed through the classifiers (after relevant pre-processing in the back-end) to identify and find the sheet metal part type (if it is a sheet metal part) or discard it otherwise.
- We can use `SQLite` as the database management agent for storing the CAD files corresponding to the identified sheet metal part type from the previous steps.
- `Flask SQLAlchemy` can be used to interact with the database to retrieve the stored CAD file.

5.4 Final Deployment

- The Web-App can be deployed on any of the popular web hosting platforms like 'Heroku' or 'GitHub pages', which can make the app accessible to users via web URL.
- We can also keep interacting with the models on the browser itself using `ONNX.js`, which is a JavaScript library. Furthermore, the app can be made more interactive via various JavaScript libraries like `jQuery` and `React.js`.

Acknowledgment

We would like to express our sincere appreciation to everyone who has contributed to the completion of this project. We would like to thank Sanjiv Doraiswamy and Wolf Wadehn for providing us with encouragement, feedback, and assistance throughout the project. We would also like to thank Nancy Kuriakose for the smooth onboarding and support throughout the project.

In addition, we would like to acknowledge Trumpf Metamation and its staff members for providing us with the necessary resources and facilities to complete this project.

References

[1] N. Neogi, et al. *Review of vision-based steel surface inspection systems*, EURASIP Journal on Image and Video Processing.

[2] A. Singh, et al., *Deep Learning-Based Defect Inspection in Sheet Metal Stamping Parts*, NUMISHEET.

[3] Yann LeCun, et al., *Object Recognition with Gradient-Based Learning*, Association for Computing Machinery.

[4] B. Liu, et al., *Automatic detection technology of surface defects on plastic products based on machine vision*, MACE

[5] Kaiming He, et al., *Deep Residual Learning for Image Recognition*, Computer Vision and Pattern Recognition

[6] I. Goodfellow, et al., *Generative Adversarial Networks*, NeurIPS

[7] A. Radford, et al., *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, ICLR

[8] X. Qin, et al., *U^2-Net: Going Deeper with Nested U-Structure for Salient Object Detection*, Pattern Recognition

[9] S. Yeung, *https://ai.stanford.edu/ syyeung/cvweb/tutorial2.html*

[10] XS Poma, et al., *DexiNed: Dense EXtreme Inception Network for Edge Detection*, Pattern Recognition