Scientific Publications

Journal of Art and Design, 2021, 1, 1331
www.scipublications.com/journal/index.php/jad
DOI: 10.31586/jad.2021.1331

Review Article

Deep Learning Applications for Computer Vision-Based Defect Detection in Car Body Paint Shops

Dwaraka Nath Kummari [1,*]

[1] Software Engineer, USA

*Correspondence: Dwaraka Nath Kummari (dwarakanathkummari@gmail.com)

Abstract: The major automated plants have produced large volumes of high-quality products at low cost by introducing various technologies, including robotics and artificial intelligence. The code of many defects on the surface of products is embedded with economic loss and sometimes functionality loss because products are rarely found with defects. Therefore, most items' production is done based on prediction and has an invisible fluctuation in production. The detection process for hidden defect images requires a lot of costs and needs to be supported for better progress and quality enhancement. Paint shop defects should be analyzed from color changes to detect defects effectively by preventing the variability of product demand over time. It is not easy to take visible light images without noise because the paint surfaces are glossy. A few parts of illumination and shadows remain in images, even in larger size and high-resolution images. The various painted surfaces are also needed to reflect both color and texture information in computer vision models to classify defects precisely. Several automated detection systems have been applied to paint shop inspections using lasers, infrared, x-ray, electrical, magnetic, and acoustic sensors. The chance of paint shop defects can be low, unnecessarily low, compared to clouds in the sky, but those chances impact defect functionalities. Thus, they are called as "lessons learned." Lately, artificial intelligence has been introduced to the field of factory automation, and many defect detection feeds have found footsteps in machine learning and deep learning. Recent attempts at deep learning-based defect detection are proposing simple techniques using specific neural network architectures with big data. However, big data is still in its early stages, and significant challenges exist in normalizing and annotating such data. To get cost-efficient and timely solutions tailored to automotive paint shops, it might be a better consideration to combine deep learning solutions with traditional computer vision and more elaborate machine learning methods.

Keywords: Automated Plants, Factory Automation, Surface Defects, Paint Shop Inspection, Robotics, Artificial Intelligence, Predictive Production, Hidden Defect Detection, Color Variability, Glossy Surfaces, Image Noise, Illumination Effects, Texture Classification, Computer Vision, Sensor Technologies, Machine Learning, Deep Learning, Neural Networks, Data Annotation, Hybrid Detection Systems

How to cite this paper:
Kummari, D. N. (2021). Deep Learning Applications for Computer Vision-Based Defect Detection in Car Body Paint Shops. *Journal of Art and Design, 1*(1), 1–18.
DOI: 10.31586/jad.2021.1331

Received: August 16, 2021
Revised: October 17, 2021
Accepted: November 12, 2021
Published: December 26, 2021

1. Introduction

In this work, we propose the use of computer vision and deep learning-based approaches for visual defect detection in car body paint shops, in particular eliminating or reducing human visual inspection in process controlling and quality assurance stages. We describe a model fine-tuning process using transfer learning that is capable of addressing any specific visual defect and/or collection of defect classes. The trained models can detect previously unseen defects in newly painted body parts and classify them into defect classes. Proper dust particle detection is done as a preliminary step to

reduce dirtiness detection false positives. We also show how models based on specific architectures can be deployed for fast inference in edge computing devices on-premise, respecting processing time constraints for production line operation. Finally, we present a set of metrics useful to assess visual detection error in comparison to human experts and discuss the advantages and drawbacks of our approach as well as future lines of work.

Automobile body painting is the last step in the manufacturing process in the car industry. It is a very complex process that should be done with care, as body defects can cause low quality, appearance, and high after-sale service costs.

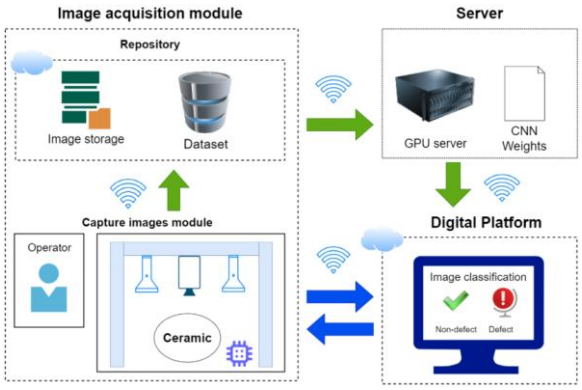

Figure 1. Automated Defect Detection System for Ceramic Pieces Manufacturing

Most paint shops have a process control and quality assurance stage where human visual inspection takes place to detect and classify defects for corrective action. This task is performed by highly experienced workers who have very refined skills. They can quickly and accurately detect the presence of any visual defect either on the fully washed and dried painted surface of the body part or on the final painted body part, where the area is frequently covered with dust particles.

1.1. Background and Significance

Automobile production, comprising metal processing, painting, and assembly, generates a wide variety of surface defects in painted car bodies, which adversely affects the aesthetic appeal desired by customers. Consequently, the detection of paint defects is crucial for quality and reliability assurance in production since the primary objective of a body shop is the delivery of defect-free painted car bodies that meet strict specifications. Defect detection is, however, a significant challenge given the size and configuration complexity of the painted car body, and the variability in the color, texture, and geometrical features of the paint surface. Large volumes of vehicles are produced at high speed in a paint shop, demanding a defect detection system with the required speed, accuracy, and reliability. Moreover, manufacturing companies require that detection be done without damaging the parts and that they be completed within their initial time goals. The painting process requires large investments and has a lot of associated costs. If defects are identified in the refining stages, costs can be reduced, and product quality enhanced. Any defect identification that may arise as a result of damage or problems during processing must be identified in real time and with high accuracy. The conventional video monitoring of painted car bodies detects but does not identify defects, which still necessitates a manual inspection. Despite the introduction of automated optoelectronic technologies for inspecting defects, it has long been a difficult and even unrewarding undertaking to design a reliable defect detection system integrated within the production line.

Automated defect detection increases the speed and accuracy of painted car body inspections by eliminating human error, supplying more comprehensive defect data, and reducing repair cycle time. However, some challenges remain, especially with the evolution of new paints and finishes that are more difficult to inspect. Artificial intelligence and machine learning approaches, especially the new trend of deep learning, show great promise in addressing the challenges. These new techniques are showing increasing progress in tackling real-world visual inspection problems, motivated by the growing available image datasets for training and large-scale advances in machine learning infrastructure. Robust machine learning algorithms proactively identify, analyze, and classify problems without producing additional closed-loop downtime.

2. Overview of Defect Detection

Visual inspection plays a vital role in the monitoring of manufactured products, assisting operators with the detection of potential defects through a visual scan. Every item has certain attributes which differentiate it from the others. Any undesired presence of these features on the object is called a defect. Product quality managers are concerned about the presence of defects in manufactured products because they detract from the appearance and functionality. All manufacturers globally are increasingly utilizing more efficient automated processes. Manufacturers are striving to realize their consistent growth in productivity, quality, and reliability to gain market competitiveness. Added to this fast-growing economy, consumers require and expect an increasing level of quality which reduces the importance of defects in manufactured products.

In the industrial production of car bodies, particular processes used to guarantee a perfect aesthetic quality are surface preparation, coating, and curing. They are usually processed without tolerance. Any defect may be a beginning interaction with the mechanism of aging of the product, and therefore detection of these defects at the next stage of production is important. The conventional methods for car body inspection systems are visually inspecting for small scratches, dust particles, or color difference by the operators, however, it is a heavy burden on them and is not only subjective but also inefficient. Automated defect detection technology is getting more and more attention because of the increased demand for quality. On the other hand, as the automated production level increases, product surface inspection is becoming one of the problems that must be solved to automate the operation of a cycle. According to the foregoing facts, this paper will propose a deep learning approach for a defect detection algorithm in a paint shop line.

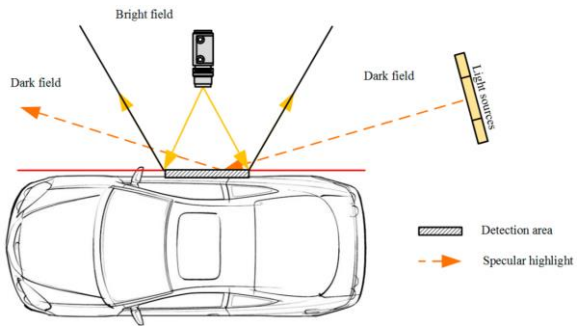

Figure 2. Surface Defect Inspection System

2.1. Importance of Defect Detection in Manufacturing

One of the most important factors in the manufacturing process is the maintenance of product quality. Quality assurance becomes one of the key factors in determining a company's competitiveness in the marketplace as a greater focus shifts from quantity to

quality. In several industries such as the automobile industry, a big loss occurs after a product is sold to customers. The loss can happen because of a certain proportion of defective products sold to customers, which results in a large reduction in a company's profit due to repair and warranty costs. Machine parts or products that are being created must be inspected during the manufacturing process to reduce the losses caused by a decrease in product quality. For this reason, defect detection is critical in manufacturing. Since the introduction of automation in factories, equipment monitoring systems have been inspecting machine parts or products automatically. Among the different equipment monitoring systems, vision systems have been used to inspect machine elements and products as they have superior capability in measuring large amounts of data without contact and at a faster speed.

The finish of the car must give satisfaction, which is why it becomes a painstaking task to carry out visual controls by hand to estimate the quality of the paint, even leading specialized teams to travel to the factory to be able to guarantee a certain quality, representing a major cost for the company. Optical inspection systems have proven to be highly effective in evaluating large surface areas, although they still have some drawbacks. The inspection is done in line with high speed, saving time, and allowing you to validate the product in front of the dealer instantly, eliminating the qualitative component that humans have; however, it is not 100% infallible, since it must be fed with the data of what is considered a normal lamp; Any fault or anomaly that occurs in the paint must be registered in order not to be rejected. Various types of faults can be detected, such as dirt or dust, which is manifested by dull dots on the surface; die stacks; particles immersed in the paint layer that produce a brighter dot; and surf dirty, which consists of a split point.

Equation 1: Pixel-Wise Defect Classification Loss

$$\mathcal{L}_{cls} = \sum_{i=1}^{N} \left[y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right) \log\left(1 - \hat{y}_i\right) \right]$$

Where:

- $\mathcal{L}_{cls}$: Binary Cross-Entropy Loss
- y_i : Ground Truth (0 or 1) at Pixel i
- $\hat{y}_i$: Predicted Probability at Pixel i
- N : Total Pixels

2.2. Common Defects in Car Body Paint

Quality in car body paint is essential since it guarantees the adequacy for use and the long-term endurance of cars. Thus, car body paint defects have to be detected and eliminated during production. Defects that are normally detected in car body paint include scratches, service blemishes, runs, orange peel, gloss shifts, and non-uniform gloss. Scratches are narrow grooves or cuts in the clearcoats or paint surface. They give an ugly appearance to the car. Service blemishes are paint or clearcoat drying defects, resulting from contamination or physical defects in the paint atmosphere, such as temperature and humidity. They can look like fine to coarse copies of dust, hair, or sand on the car's surface. In paint coating, runs, streaking, or sagging are defects characterized by the thickening, bulging, and distortion of paint systems around edges on the surface. Orange peel is a surface texture defect, producing a non-uniform surface gloss. It looks like the peel of an orange, characterized by shallow, irregular bumps and recesses on the surface. Gloss shifts are variations in surface gloss within the same color that are usually associated with

the orange peel or wetting behavior. Non-uniform gloss is the name given to instrumentally determined variations in surface gloss for a given color during gloss measurement. The defects that affect car body paint after delivery because the repair is expensive and difficult are scratches, runs, gloss shifts, orange peel, non-uniform gloss, and service blemishes.

3. Deep Learning Fundamentals

In this chapter, the basic principles of DNNs are discussed, which are important to understand the transfer learning technique installed in the second part of the application development. Readers already familiar with the basic principles of DNNs can skip this chapter. However, we took care to avoid overly technical explanations so that this chapter can be accessible even to readers without an in-depth understanding of DNNs.

1. Neural Networks Overview

Artificial Neural Networks (ANNs) consist of interconnected artificial neurons using weighted connections. Each connection has an associated numerical weight. The input to an artificial neuron is the weighted sum of the inputs from the previous layer plus a bias term that is specific to that neuron. The output of the neuron is obtained by passing the computed input through an activation function (usually nonlinear). Each neuron computes the distribution of inputs represented by the corresponding weights and bias and activates (outputs confidence) based on the chosen activation function. DNNs, layers of artificial neurons, are then trained using a supervised learning method based on example input-output samples. In supervised learning, the network is fed with input patterns, and the output of the network is compared with the desired output target, and the difference is used to update the connection weights by the backpropagation learning algorithm.

DNNs provide a powerful model for nonlinear regression and classification problem types. They have been used for many applications such as image recognition, speech recognition, fraud detection, and language translation. DNNs possess important characteristics that enable them to deal with complex regressions or difficult reasoning tasks. Specifically, a DNN is able to perform hierarchical feature extraction of the input data that can then be invoked for classification or decision-making. They represent a similar approach to how humans think [1].

3.1. Neural Networks Overview

Neural Networks (NNs) are inspired by the structure of biological neural networks and were developed to recognize patterns in the early 1960's. Pattern recognition is the classification of inputs into different classes - a set of known categories. While NNs are based on biologically inspired computation models, they are much less mathematically complicated than the models used to simulate brain function. However, unlike mathematical brain models, NNs require considerable amounts of data for training as well as considerable computational power. Neural networks comprise an input layer, an output layer, and usually at least one hidden layer. The inputs are in the form of a large number of features of the data of interest. The post-processing steps in an NN transform the raw input features at the input layer to the final outputs at the output layer using NNs with a weighted summation followed by a nonlinear activation function.

An NN comprises neurons in an array-like organization, with each neuron connected to neighboring neurons, as well as input and output connections. The outputs of each layer of neurons are determined by hyperbolic tangent activation functions (or other nonlinear functions) that respond to the linearly combined inputs with weights that connect neighboring neurons. NNs require tuning of their weights, which is usually accomplished by a supervised learning algorithm known as backpropagation. The backpropagation approach requires the availability of tagged data for algorithm learning, using tagged retirement data, and testing using independent data. Classifying the

retirement data involves presenting the NN with the inputs, for which corresponding outputs are known, and comparing the actual with the expected outputs.

The NN adjusts its weights to minimize the difference between actual and expected. This is done using one of several algorithms, an example being the gradient descent method. The performance of the NN will depend on the quality of data, the complexity of the problem, the number of training points required for convergence, and the optimization of hyperparameters. Typical examples of NN applications are in speech and image recognition.

3.2. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) were launched in the mid-1980s, gathering inspiration from biological phenomena. It is recognized that scaling and translation invariance are necessary when utilizing deep learning to process computer vision data, including images and videos. CNNs offer specific architectural features and depth that facilitate automatic feature extraction in the case of images, establishing the components in the right order relative to the surface normal of the object being recognized. CNNs can deal internally with translation invariance by stacking several hidden layers; specifically, one or more convolutional layers, then several subsampling layers, and a few fully-connected layers at the end. CNNs could extract hierarchical features. They can learn global features as well as local features in images. CNNs deal effectively with the curse of dimensionality thanks to their architectures.

The hidden layers of CNNs comprise convolutional layers, pooling layers, and fully-connected layers. Convolutional layers deal with data by convolving the data with kernels, also called filters. The kernels of a convolutional layer can be seen as ordered sets of parameters that are adjusted through training to minimize empirical loss on training data, so they have the same function as the weights of a fully connected layer. The thing that distinguishes kernels in a convolutional layer is that they are applied locally to the input and are always the same, thanks to weight sharing. Pooling layers subsample the data. In practice, they shrink the 2D grids that the convolutional layers feed them with, so that the following layer has a smaller input size. Their main effect is that they gradually become translation-invariant, thereby enabling the network to use deeper architectures. Fully-connected layers read all the inputs and connect them to all the outputs, so they are the densest layers. However, fully connected layers are not generally used today in the case of images, except for the final layers at the end of the architecture.

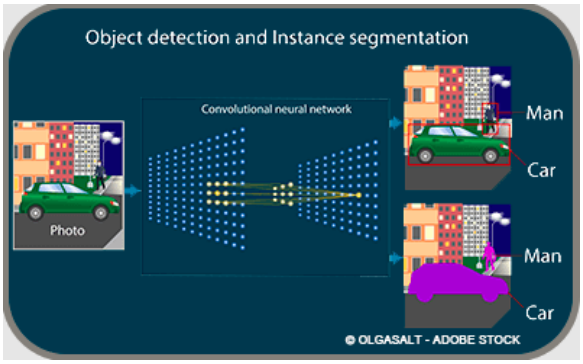

Figure 3. Convolutional Neural Network

3.3. Transfer Learning in Deep Learning

Deep Learning Methods have achieved significant success in many domains due to the availability of large amounts of training data, vast computational resources, and carefully crafted model architecture designs. These factors have driven the performance of many applications to near-human performance levels. However, the growing data

demands of Deep Learning Models along with the prohibitive costs of training them have driven researchers and industry practitioners towards Transfer Learning. Transfer Learning aims to overcome the limitations of high training costs by re-using a pre-trained model that has been trained on a different but related task.

In its simplest form, Transfer Learning involves reusing the model weights of a model that has been pre-trained on a large dataset of related tasks. The pre-trained model could be either a model that has been trained on real-world data for the target problem or a model that has been trained on one of the benchmark datasets. Models initialized with weights from a similar task often converge significantly faster compared to models being trained from scratch. The transfer approach has become the standard approach to model training for many computer vision tasks. The majority of the capable architectures in use today perform Transfer Learning in one of two ways. In the first approach, only the last Fully Connected layer is re-trained, while the CNN is used for feature extraction. Features obtained via a pre-trained network train a simple classifier such as an SVM, which has been shown to successfully converge. The second approach retains the model architecture while re-training all the model layers on the target problem. The second approach allows the Shared CNN to learn more discriminating features due to task specialization. Both methods involve the re-training of a model for the target dataset, allowing the importance of task specialization to dictate the final practical performance.

4. Computer Vision Techniques

Over the past 30 years, image processing, machine learning, and especially Deep Learning techniques have occupied a central place in the rapidly gaining computer vision research field. Computer vision is an interdisciplinary field of computer science, engineering, and applied mathematics. It enables electronic devices or robots to automatically analyze and understand visual data from the world around them and make decisions and predictions based on this information. In recent years, with the development of computational capacity, a huge number of advanced image analysis and recognition techniques have been developed and mathematically formalized such as image filtering, feature extraction, object segmentation, object matching, and analysis of motion. In this work, we review the fundamentals of image processing and some DL-based modern vision techniques.

Image processing is a physical, chemical, and mathematical process in which an image is quantitatively classified and/or qualitatively changed. Digital image processing uses computer methods and algorithms to overcome the limitations of optical images. A digital image is a set of pixels and intensity values. Digital image processing plays an important role in image storage, transmission, enhancement, segmentation, feature extraction, recognition, and interpretation. A key function of image processing is to compress an image into a small number of features, whether in analysis or classification. There are several image transformation techniques available for image preprocessing, noise removal, filtering, compression, and segmentation. After images have been processed, known analysis techniques are used to extract valuable features from images. Feature extraction greatly simplifies the data analysis step but requires the knowledge of prior features.

4.1. Image Processing Basics

Digital images consist of pixels arranged in grid form. Each pixel contains components representing the intensities of light of given wavelengths. The overall values create a digital representation of a visual scene. Digital images can usually be represented as three-dimensional structures with x- and y-axes representing the two spatial directions of the image. One image property is the range of intensity values. For grayscale images, each pixel has a scalar value representing its luminance, while color images usually contain three channels representing the intensities of given wavelengths, typically red,

green, and blue. The three channels can be stacked to form a three-dimensional structure. Image processing techniques for computing a desired output image from an input image usually operate on a pixel basis. Image operators that can be applied to an individual pixel in isolation include thresholding and point operations such as gamma correction, Log, and exponential transformations. Thresholding is fundamental in image processing and involves classifying pixels based on whether they are above or below a certain value. For instance, in the case of grayscale images, all pixel values below a certain threshold can be made black, while pixels above that threshold can be made white. Color thresholding involves creating binary maps for each channel using center values and determining a pixel's final classification based on the results. Furthermore, image smoothing techniques emphasize the local neighborhood of image pixels and can be implemented using averaging filters. These pixel-based techniques for brightness or color transformations are powerful in any image processing task, including the detection and characterization of painted surfaces in a car body paint shop [2].

4.2. Feature Extraction Methods

Image Feature Extraction is an essential and initial stage in machine vision applications such as image detection, segmentation, classification, and recognition. The region of interest in almost every image processing problem will have unique features but not all may be used for image formation and recognition. Though thousands of local point features may be extracted from an image using various techniques, only a few global features based on color, edge, texture, motion, shape, volume, and illumination will be useful for image recognition and classification. The Geometric Transformation and neural model-based techniques will extract rich color and shape orientation features from an image but these are expensive and not all the time accurate. The model- and neural-model-based object detection and recognition techniques using object model matching in images are potentially very accurate but computationally very expensive. Even if a few explicit constraints are given, these methods do not always find a solution.

Different features contribute different information for the object recognition stage. Therefore, the combination of two or more features such as color and shape of the object, shape and texture of the object, edge orientation histogram, color, and motion of the object, point, line, and surface, etc. is widely used in feature-based object/image detection and recognition systems. The use of knowledge about the placement, functionality, and interaction is essential as it constrains the model and has a significant impact on the performance of the overall system. The construction of a color or motion-based hierarchical scene classification from interactions to ease the retrieval of the right object models is very necessary to avoid model clashes in recognition. The prior knowledge combined with the hierarchical feature language model provides a useful constraint and also constitutes a higher-order feature-based object detection and recognition [3].

5. Data Acquisition and Preparation

In this chapter, we will explore data acquisition and preparation methods, as we need a large amount of labeled image data to train the DL models, and the image data from real-world applications are usually unbalanced and lacking in defect variety. The pipeline of image data preparation for DL-based DDD is specified here. We collect the data from existing datasets or some other sources. The image data are preprocessed, to mix them up properly and then to be fed into a DL model. Then, we describe several important methods to prepare image datasets effectively. Data preparation plays a pivotal role in the development of the DDD models because the performance of these models heavily relies on the quality of the prepared datasets. The quality of an image dataset is dependent on not only the number of images but also the number of defective images; moreover, there should be a visible difference in the features of defective images. In single-source data databases, datasets exhibiting a shortage of defective image diversity and an imbalance

between the numbers of various defective classes can lead to performance degradation. Therefore, techniques such as data annotation and image data augmentation are vital to the effective utilization of image datasets.

In conventional image DDD research, pixel-level labeling, such as the masking of defects, has always been the standard for supervised training. Additionally, large-scale datasets for supervised training require pixel-level labeling, a tedious and time-consuming process. To resolve this issue, researchers opt for weakly-supervised learning approaches. Domain adaptation is a solution to overcome the lack of labeled image data. A pipeline for automated paint defect detection based on FBSD for fine-tuning models is proposed.

Equation 2: Intersection-over-Union (IoU) for Defect Localization

$$IoU = \frac{A_{pred} \bigcup A_{true}}{A_{pred} \bigcap A_{true}}$$

Where:

- A_{pred} : Predicted Defect Area
- A_{true} : Ground Truth Defect Area

5.1. Collecting Image Data

Collecting representative training data is crucial in training a machine learning model, as the quality of the data put into the model during training will determine how well the model generalizes and performs on any unseen data. Without realistic, domain-specific data, there is a high chance of incurring a decline in model performance. In the automotive domain, defects that occur during production are each categorized into defect types. Each defect type may have multiple unique representations or variations. Because visualizing the data incurred during production for each defect type is too difficult for an employee, storing and collecting this data is an easier approach. Storing a representative set of defect images, organized by defect type, should serve as the first point in training a deep learning model. However, there might not be enough data from the defect images to use alone due to a skewed number of unique representations of each defect type. In this case, a few images that simulate the defect type in an idealistic manner using original or clean images of the painted surface can be employed to supplement the data.

It should be highlighted that while it may take a very long time to detect defects once the product is completed, storing images from the early detection phases would enable a model to generalize better on unseen data and allow the production process to avoid proceeding further with damaging vehicles that have defects. Both model training and actual inference can benefit from this addition [4].

5.2. Data Annotation Techniques

The need for labeled datasets for training a supervised model is a critical step in the deep learning pipeline. Automated image annotation methods have matured significantly over the past few years. Most tasks in vision have some umbrella strategy for annotation/model training that can be easily adapted to the specific goal. For example, an object detection model trained on a general dataset can be used to produce bounding boxes in any domain but then needs to be refined through a user interface where the user corrects these labels. By doing this, the user will not need to label every image, and data annotation will be easier to scale. Another prevalent use case is using weakly labeled data

- for instance, image-level labels - to pre-train a pixel-wise supervised model. An example of this is using a classifier to predict pixel-wise labels for a target domain.

Coaches will also check these models. The advantage of these methods is that they are mostly free. However, there are drawbacks to gas-phase and LRF detection. Both of these defect types are usually small, complex, and challenging to detect, leading to a significant drop in performance of the weak supervision methods. For aerial LRF detection, although weak labels allow user-based correction to routinely create pixel-level noisy gold standard labels, the performance of different models depends a lot on the selection of weak supervision methods. Also, weak products are really sensitive to noise, especially gas-phase defects.

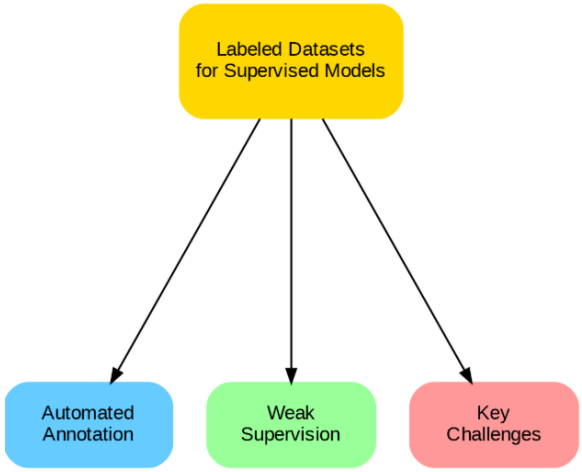

Figure 4. Labeled Datasets \ for Supervised Models

5.3. Data Augmentation Strategies

Deep learning requires vast amounts of data to train and validate neural networks due to its ability to learn highly complex functions. Moreover, the use of complex architectures, like convolutional neural networks for feature representation, demands a lot of data. In practice, having a large amount of labeled data is a difficult task. Annotating data is expensive since usually an expert on the task has to be hired. Also, in many applications, like car paint defect detection, it would be difficult to allocate money for this specific task. Therefore, data augmentation techniques are used to create new data from the original ones.

Data augmentation approaches can be roughly divided into two categories: (1) basic, which are considered traditional image processing techniques but can be used with deep learning, and (2) advanced, which refers to synthetic or quasi-synthetic data derived from Generative Adversarial Networks. Basic approaches usually use simple transformations: rotation, reflection, cropping, zooming, scaling, changing brightness, noise addition, a mixture of color and texture, random erasing, occlusion methods, and a large number of other techniques. Data augmentation algorithms reduce overfitting since they approximately generate new data for training models or fine-tune pre-trained models using the random transforms performed. In a nutshell, augmentation strategies induce a regularization effect by exploring the data manifold. However, data augmentation does not solve the issue of domain shift.

Uses of GANs to generate new data in an unsupervised way based on real images have been extensively reported, as they can transfer distribution from one domain to another. Moreover, GANs are highly customizable and allow for generating multimodal output by using several labels. In particular, the style-based GAN architecture has received important attention from the community. The main advantage of this model is

the properties learned at the intermediate latent space, which encode important variations of the data. The proposed modifications enable the characteristics of the latent space learned to generate more realistic patterns during the synthesis of the 3D data [5].

6. Model Development

Deep learning applications have recently achieved great success in data-heavy, complex tasks such as image classification, natural language processing, and image synthesis. For the specific problem of defect detection in an industrial manufacturing setting, however, the data requirements for deep learning methods are troublesome. Because video and image data from car body paint shops is plentiful and needs to be reviewed frequently, we can extract candidate patches of painted car body images that contain interesting content. We can then label the candidate patches and conduct domain transfer, fine-tuning pre-trained models on our labeled datasets. This allows us to create a model that can generalize well and is adapted to our industrial paint shop environment. In this chapter, we will discuss how we selected suitable architectures and datasets and conducted domain transfer and model fine-tuning using a combination of techniques such as data augmentation, hyperparameter tuning, and pre-trained models, among others. We will discuss how we tuned various model architectures, including Vision Transformers and Convolutional Neural Networks-based models to improve model performance for detection of a variety of dataset-specific defects. The model will be evaluated based on various metrics including confusion matrix, accuracy, loss, precision, and recall and their respective implications on the problem of defect detection. The remainder of this chapter will be organized as follows. In this section, we will discuss the model architecture selection process. In this section, we will discuss how the data augmentation and hyperparameter tuning choices and others help improve the model performance. Finally, in this section, we will discuss the various methods of model evaluation that can be used for defect classification [6].

6.1. Choosing the Right Architecture

Computer vision is one of the most popular applications of deep learning, to that extent many such works have been shared and presented online. This includes multiple models ranging from small inference-capable networks running on smaller embedded processors to heavier complex models. It can get very overwhelming for a starting IPA intervention to find the right model architecture and technology to use for the defined task and therefore we present our findings here.

All models can perform reasonably well if used and trained with the right data, the main difference between each architecture would be the accuracy and the inference speed. For an Ideal edge-device monitoring solution, inference speed would be one of the biggest priorities for the model, meaning the device needs to be able to perform the visual inspection in real-time, and therefore the model needs to be tiny enough to run on such devices. For this, we used a model known for its speed. Another important factor when selecting a model would be the accuracy. For this, we used models known to be one of the most state-of-the-art and have been shown to produce great computation results for detecting and segmenting objects from images.

Then finally, other things to keep in consideration during the model evaluation step would be the model size, overall training time, detection capability during inference time, and if the selected model is able to perform semi-supervised learning or not. And with all this in mind, we decided to use a model along with another for our particular task as we found they achieved a good performance number when evaluated towards the said criteria.

6.2. Training the Model

Training a neural network model is much like an iterative game of guess and check, done with the help of an optimizer. Initial random weight guesses are used on a training set of input data, producing outputs that are likely poor approximations of the training data's true labels. An optimizer uses a loss function to measure the model output's deviation from the true labels; if the deviation is higher than what can be considered acceptable, the optimizer adjusts the weights based on the algorithm and values of the respective hyperparameters. This process is repeated after continuously feeding the model with batches of data taken from the training set.

The method used to feed input images and their true labels to the model is referred to as a data loader. A data loader will shuffle the data randomly so that corresponding samples are not input to the model in the same order every time. Larger initial batch sizes are faster to run through the model than smaller ones, as they allow the GPU to take full advantage of its parallel processing capabilities, but later iterations of the training process usually benefit from smaller batch sizes. This is because the model is then less overly influenced by the gradient descent adjustments computed using large batches, which can overshoot the parameter values that yield minimal loss. Overfitting is a phenomenon that can occur when a model trained on sample training data produces very poor performance on validation and test data that is from the same data distribution as the training samples. To mitigate this risk, we augmented the dataset to introduce slight changes to the training, validation, and test samples. Normal and defective images were mirrored, enhanced with different brightness levels, and rotated [7].

6.3. Hyperparameter Tuning

When training a deep learning model, hyperparameter tuning is the process of improving its performance by adjusting the hyperparameters. Before looking for methods to efficiently tune hyperparameters, it might help to have a clear understanding of what they are. Hyperparameters are parameters that are not learned from the data during training. Instead, while such parameters control the training process and influence the model's architecture or representational capacity, the developer has to set them before the start of the optimization process. The developer generally sets these parameters based on heuristics and empirical results.

Learning rate is another hyperparameter, which can be thought of as the pace of the learning process. By controlling the step size in the gradient descent algorithm that updates the model parameters, the learning rate influences how quickly or slowly a network converges to an optimal solution. If the learning rate is set too high, the convergence will overshoot optimal solutions and cause the final solution to have large errors. If the learning rate is set too low, optimization may take too long, or the algorithm may get stuck at suboptimal solutions, especially if the cost function is not convex.

Dropout is a regularization method that involves randomly dropping out a certain fraction of the hidden neurons in a given layer during training. The fraction of the neurons to drop out is referred to as the dropout rate, which is typically set in the range of 0.1 to 0.5 for the fully connected layers. A very small dropout rate may not give enough regularization given that overfitting can occur in models with a high representational capacity. A very large dropout rate may result in network underfit given that too many neurons are deactivated [8].

7. Evaluation Metrics

Evaluating the performance of a defect detection model is an important step after training. There are a variety of metrics available to determine how well a network is performing. The most popular ones are accuracy, precision, recall, F1-score, and ROC AUC. Some deep learning models do output probabilities for given classes, while others will output classes directly. If a model outputs probabilities, then a threshold also needs to be selected. Selecting a good threshold is crucial: a too-small threshold will increase the

False Positive (FP) rate, while a too-large threshold will increase the False Negative (FN) rate.

1. **Accuracy and Precision:** Accuracy is the most common and simplest metric. It represents how many instances were correctly classified. However, it is usually not a good indicator for small object detection tasks, such as defect detection. In a car paint shop, the probability of a class being "normal" is much larger than the probability of any class being "defective". So it could easily happen that a network performs well on the "normal" class but badly on the "defect" class and still yields a very high accuracy. Thus, precision is commonly preferred, especially for detection tasks. Precision represents how many of the classified defects were correctly classified. However, for defect detection tasks, it is still better to look at other metrics because high precision with low recall would indicate that most defects are not detected.
2. **Recall and F1 Score:** The recall represents how many defects were detected correctly. In defect detection, we'd like a high recall with an equally high precision. However, it is often the case that we can only optimize for one of the two metrics with a selection of the threshold. The F1 score unifies both metrics by calculating the harmonic mean of precision and recall and is a more suitable metric for defect detection tasks. It is common to select a threshold to optimize for the desired metric, as networks are generally optimized based on the loss function.

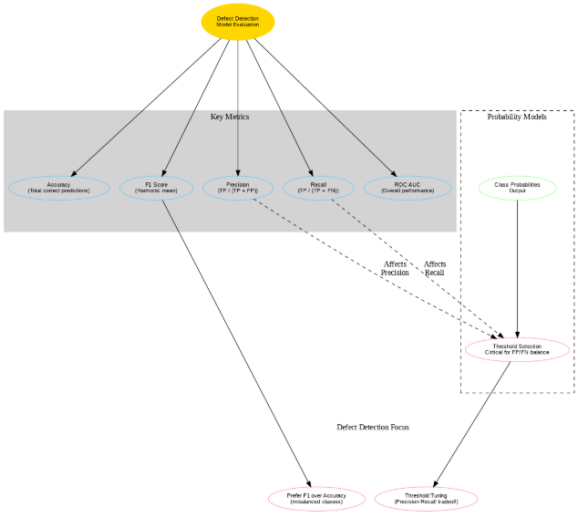

Figure 5. Defect Detection Model Evaluation

7.1. Accuracy and Precision

The evaluation of computer vision models is a complex topic that deserves in-depth discussion. Generally speaking, evaluation is needed for three main purposes: to assess the model performance, to increase the performance through hyperparameter tuning, and to determine when to stop training when techniques such as early stopping are applied.

In classification algorithms, a basic way to determine how well a model is doing is the accuracy, defined as the number of true predictions divided by the total predictions. The accuracy has drawbacks since it does not assess how other predictions are doing, meaning that it is possible to have a high accuracy while at the same time a low precision on a class of interest, for example. Precision is a measure of how good the classifier is in predicting positive results defined as the true positives divided by the sum of true

positives and false positives, meaning that it is only dependent on the positive class predictions. This is an important metric to consider when wanting a scarce class of interest to have fewer false positives since you are only interested in the positive prediction quality. One drawback of the precision measure is that if the model predicts that no samples belong to the positive class it is undefined.

It is also possible to define a macro precision that calculates the precision for each class of interest with an equal sample weighting. However, consider that when dealing with unbalanced samples between classes, assessing all classes equally is not a good idea and one must pay attention to the weightings of the samples. There is also a micro precision defined by the sum of the true positives for all classes divided by the sum of the true positives for all classes and the sum of the false positives for all classes. This is a valid metric in unbalanced problems because it does not weigh equally all the classes when determining how well the model is doing [9].

7.2. Recall and F1 Score

Another issue with the Precision metric is that a high value of the Precision does not indicate a high performance of a model. For example, say we have a dataset of 100 samples, out of which 1 is a "positive" sample and 99 are "negative" samples. Now, if we create a model that classifies every sample as a negative sample, the Precision would be equal to 0 and we would have no False Positive Rate. But is this good? No, because we cannot ignore the one positive sample in the dataset. So, we need a metric that takes care of both the value of True Positives as well as the False Negatives. To do this, we need to define a new metric called Recall.

The Recall metric is mathematically defined as the ratio of True Positive samples to the total number of Positive samples in the dataset. In other words, the Recall indicates how many actual positive samples that were included in the original dataset were correctly classified as Positive by the model. The downside for the Recall metric is that it only checks for the False Negatives and works on that factor only. But to overcome that, we can combine both Precision and Recall and calculate a metric called the F1 Score. The F1 Score takes the two metrics into account and checks both True Positive as well as False Negative samples while combining both metrics. It is given as the harmonic mean of both Recall and Precision metrics. Therefore, F1 Score is usually a good metric to evaluate the performance of a model especially in cases where the classes in the dataset are unbalanced [10].

7.3. Confusion Matrix Analysis

Deep learning models are frequently used in defect detection tasks where the majority of data points are classified as ‘normal’ images, while only a fraction belong to the defect classes. The high class imbalance during training results in straightforward metrics such as accuracy to fail in conveying the performance levels of the models. The confusion matrix illustrates the performance of the model on predicted labels in comparison to the true labels of the test dataset. By calculating the number of true positives, true negatives, and classification errors for all the various defect types and the ‘normal’ label, it presents a compact but efficient illustration of the various model performances. The ideal is definitely to be ‘green’ in all defects detected. However, due to the nature of car body assembly lines, not all defects can be detected in all the car bodies resulting in certain defects being only dimly illuminated or difficult to capture. Therefore, certain defects may result in a higher level of confusion during classification.

The plots generated by the confusion matrix represent global classification performances for color mapping, in that both color and brightness levels reflect the percentage distribution of detection performance in the misclassified image. From the matrix, it can be seen that certain class predictions perform better for certain defects and are aware of the underlying fact that not all defects can be correctly detected for an

individual image. Additionally, a true 'normal' image where no defect is present is available for all the car bodies, including the images for the various defects. All these facts have been incorporated in the post-processing stage for false positive rejection and negative relabeling. The analysis provides important guidelines for further data collection and annotation designs as well.

8. Deployment Strategies

The most time spent in a manufacturing process, from development to final production, is called deployment and the systems must be integrated with software already in production. Any decisions taken as part of the manufacturing planning and system design dictate the final efficiency reached and such decisions are considered the realistic data obtained during the validation of the project, mainly during the pilot validation phase. For recruited production systems, a full-gain participating in avoiding defect will be intended, each zone is defined, each percentage of utilization will be consulted, and then the deviations by conditions will become the instance of attribute tolerance or the defining quantity of each class. The functionality of a paint shop can be defined by the capacity of the painting system, which is calculated based on the production units needed to lose the total time of the order after total located allocation. The path of a vehicle through a paint shop is generally defined by the number of cavities throughout the time allotted for completing the vehicle, from entry to exit from the phase where the coating is cured; also, the local configuration must minimize the transit time in the booth conveyances and maximize the area for which a unit remains stationary. A quick and reliable detection of defects will be essential to establish the related production strategies, increasing speeds, product throughput, and decreasing cycle times. If the decision is taken to detect all possible defects, the detection speed must be minimized, and the detection time must be maximized; however, the production strategies chosen should additionally optimize the processes before painting, especially the electrostatic treatment of the vehicle surface [11].

8.1. Integration with Manufacturing Systems

Computers have been integral to all stages of the paint shop process since the advent of automated process control systems. Systems are already in place to monitor every aspect of the manufacturing process, such as conveyor speed, temperatures, humidity, and cleanliness, among others. These parameters are stored and available for CRM purposes. The computer-controlled systems thus have the potential to store the motivation of defects. Given these capabilities, it is sensible that a neural defect detection capability could be integrated as part of the process. Defect detection capabilities have been required for several years and currently, various methods are employed to both monitor quality and act on a defect. Some key aspects need to be taken into account when integrating neural methods with manufacturing systems.

Any defect detection implementation system should provide feedback about decisions taken so that operators and the computer-assisted decision-making system can act accordingly. The feedback could either be on how many defects are detected, the processing time taken to detect and validate a defect, or the probability a defect will take place. This allows for supervising the process. This system has to be implemented as part of a decision-making module. The feedback is also required by organizing the paint shop system to know precisely where the defect is taking place. Neural networks are criticized because they are easy-to-use black-boxes that do not provide any type of feedback. Recent advancements in optimization techniques and graphics are attempting to address it. Probabilistic graphical modeling methods extend to probabilistic feedback capabilities and have already shown promising results.

Integration towards automated systems is important not only because of how long manufacturing takes to change but also because issues such as part handling, defect

checking, fixing, and idiosyncratic solutions impact logistics and the cost of the painted product. Manufacturers and suppliers are aware of each other and there have been from time to time contacts between manufacturers of automotive paint and manufacturers of paint booths.

Equation 3: Confidence-Weighted Detection Score

$$S_d = \alpha \cdot C_d + \beta \cdot IoU$$

Where:

- S_d : Detection Score
- C_d : Model Confidence Score
- IoU : Localization Accuracy
- α, β : Weighting Factors ($\alpha + \beta = 1$)

8.2. Real-time Defect Detection

Car manufacturers typically strive to ensure that their paintwork meets the strictest quality requirements. Some platforms are known for their critical defect identification. At the same time, the end customer is sensitive to the defects, which may affect the reputation of car manufacturers. Furthermore, color-mismatching defects are the most critical ones, and especially defects located at such critical areas, need to be inspected, because they are visually freelancing the vehicle, and often these defects are detected by the customer at the final inspection point, which results in high costs for the manufacturers. Nevertheless, due to the complexity of color-matching defect identification, and the typical paintshop settings, it is difficult to realize an automatic defect detection with sufficient quality. This has led several car manufacturers to introduce manual inspections of color-matching defects using complex inspection plans, which define the inspection schedules and detailed inspection procedures.

Manual visual quality inspections are still the most common actual check for surface defects, but are expensive and slow and suffer from fatigue. Thus, manual inspection is often performed only at the end of the process chain. Consequently, further processing operations, as well as delivery of the product, are done with already known product defects and often risk extremely high financial implications. In contrast, an automated inspection system is preferred from a theoretical point of view, as it provides solutions for all of the above problems. Using advanced state-of-the-art deep learning techniques, which usually achieve promising results, it is possible to reduce the number of false positives and false negatives. However, these methods require sensitive tuning for every specific application [12].

9. Conclusion

Deep learning algorithms have provided state-of-the-art performance for various computer vision applications. Their outstanding performance has made them an attractive solution for defect detection in car body paint shops as well. However, to date, such state-of-the-art deep learning methods have not been sufficiently exploited in the field of defect detection for car body paint shops. In addition, such defect detection tasks come with severe restrictions on the size and form of the available training data sets, that is, the training data consists of only a limited number of images for each defect class due to the rarity and high cost of defects during the training phase. We reviewed the challenges involved when transferring fully automated defect detection solutions into car

body paint shops and gave an overview of our predominant research results to meet the requirements of OEM and Tier-1 manufacturers.

We started with a novel defect segmentation task and presented a concept that exploits multi-instance learning using the fully convolutional model as a segmentation baseline. We then discussed topic-specific modifications to the fully convolutional approach based on our proposed segmentation model. Next, we continued with a multi-label classification task that focused on human-in-the-loop concepts. These two core topics of our research indicate that we cover both the fully automated and the assisted-defect-detection tasks. A fully automated solution can only be successful if the results are of sufficiently high quality. Otherwise, the OEM and Tier-1 manufacturers of car body paint shops will have to rely on human assistance during the defect detection process. We concluded with some pointers for future research directions, including new developments for low-data scenarios and a reliable estimate of defect presence and severity for a fully automated solution [13].

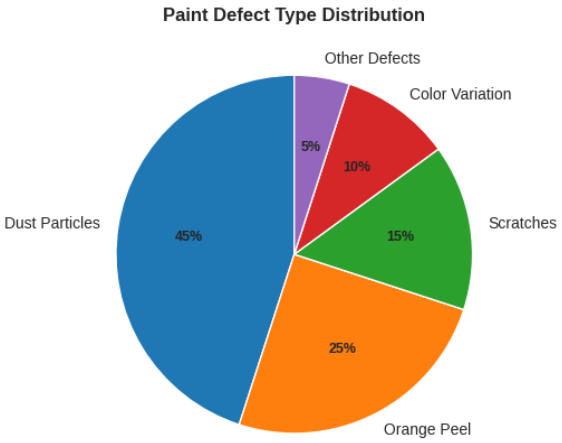

Figure 6. Paint Defect Type Distribution

9.1. Final Thoughts and Future Directions

This chapter closes by summarizing the main contributions of the thesis. First, the dependencies of the car body paint shop quality counseling were settled in terms of visual quality review. Then, a systematic investigation of various deep learning-based defect detection techniques has been presented, with a focus on the importance of data preparation and annotating the dataset, which required the use of separate annotating tools depending on the type of defect class. In addition, several data augmentation methods were adopted to train the models to deal with a limited dataset size. Subsequently, different classical machine learning methods were reviewed, calibrated, defined, and implemented, and through experimental determination, established the range of classical methods. Based on these findings, novel combined models were developed and successfully applied to real paint shop data. Finally, a thorough model performance analysis was performed on dataset samples, using various metrics, and applying specific performance criteria that the defect detection model would need to satisfy.

The presented model outperformed existing studies since previously reported solutions did not consider defect classification accuracy. The work was placed under demanding conditions required by the real paint shop environment, and the established defect detection model was tuned using real samples after initial development on a virtual dataset and was proven to be capable of specific defect type training using limited training dataset size. This ability was further demonstrated using the model on actual production cars. Concerning industrial adoption, the proposed model can be integrated with existing

optical visual inspection systems since it operates fully automated without the need for human interference. Therefore, the mapping of pixel-wise defect classes with transfer maps sets of painted car body volumes could finally provide support for paint shop operators [14].

References

[1] Nuka, S. T., Annapareddy, V. N., Koppolu, H. K. R., & Kannan, S. (2021). Advancements in Smart Medical and Industrial Devices: Enhancing Efficiency and Connectivity with High-Speed Telecom Networks. Open Journal of Medical Sciences, 1(1), 55–72. Retrieved from https://www.scipublications.com/journal/index.php/ojms/article/view/1295

[2] Chava, K., Chakilam, C., Suura, S. R., & Recharla, M. (2021). Advancing Healthcare Innovation in 2021: Integrating AI, Digital Health Technologies, and Precision Medicine for Improved Patient Outcomes. Global Journal of Medical Case Reports, 1(1), 29–41. Retrieved from https://www.scipublications.com/journal/index.php/gjmcr/article/view/1294

[3] Avinash Pamisetty. (2021). A comparative study of cloud platforms for scalable infrastructure in food distribution supply chains. Journal of International Crisis and Risk Communication Research, 68–86. Retrieved from https://jicrcr.com/index.php/jicrcr/article/view/2980

[4] Anil Lokesh Gadi. (2021). The Future of Automotive Mobility: Integrating Cloud-Based Connected Services for Sustainable and Autonomous Transportation. International Journal on Recent and Innovation Trends in Computing and Communication, 9(12), 179–187. Retrieved from https://ijritcc.org/index.php/ijritcc/article/view/11557

[5] Balaji Adusupalli. (2021). Multi-Agent Advisory Networks: Redefining Insurance Consulting with Collaborative Agentic AI Systems. Journal of International Crisis and Risk Communication Research, 45–67. Retrieved from https://jicrcr.com/index.php/jicrcr/article/view/2969

[6] Singireddy, J., Dodda, A., Burugulla, J. K. R., Paleti, S., & Challa, K. (2021). Innovative Financial Technologies: Strengthening Compliance, Secure Transactions, and Intelligent Advisory Systems Through AI-Driven Automation and Scalable Data Architectures. Universal Journal of Finance and Economics, 1(1), 123–143. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1298

[7] Adusupalli, B., Singireddy, S., Sriram, H. K., Kaulwar, P. K., & Malempati, M. (2021). Revolutionizing Risk Assessment and Financial Ecosystems with Smart Automation, Secure Digital Solutions, and Advanced Analytical Frameworks. Universal Journal of Finance and Economics, 1(1), 101–122. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1297

[8] Gadi, A. L., Kannan, S., Nandan, B. P., Komaragiri, V. B., & Singireddy, S. (2021). Advanced Computational Technologies in Vehicle Production, Digital Connectivity, and Sustainable Transportation: Innovations in Intelligent Systems, Eco-Friendly Manufacturing, and Financial Optimization. Universal Journal of Finance and Economics, 1(1), 87–100. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1296

[9] Cloud Native Architecture for Scalable Fintech Applications with Real Time Payments. (2021). International Journal of Engineering and Computer Science, 10(12), 25501-25515. https://doi.org/10.18535/ijecs.v10i12.4654

[10] Pallav Kumar Kaulwar. (2021). From Code to Counsel: Deep Learning and Data Engineering Synergy for Intelligent Tax Strategy Generation. Journal of International Crisis and Risk Communication Research, 1–20. Retrieved from https://jicrcr.com/index.php/jicrcr/article/view/2967

[11] Chinta, P. C. R., & Katnapally, N. (2021). Neural Network-Based Risk Assessment for Cybersecurity in Big Data-Oriented ERP Infrastructures. Neural Network-Based Risk Assessment for Cybersecurity in Big Data-Oriented ERP Infrastructures.

[12] Katnapally, N., Chinta, P. C. R., Routhu, K. K., Velaga, V., Bodepudi, V., & Karaka, L. M. (2021). Leveraging Big Data Analytics and Machine Learning Techniques for Sentiment Analysis of Amazon Product Reviews in Business Insights. American Journal of Computing and Engineering, 4(2), 35-51.

[13] Routhu, K., Bodepudi, V., Jha, K. M., & Chinta, P. C. R. (2020). A Deep Learning Architectures for Enhancing Cyber Security Protocols in Big Data Integrated ERP Systems. Available at SSRN 5102662.

[14] Chinta, P. C. R., & Karaka, L. M.(2020). AGENTIC AI AND REINFORCEMENT LEARNING: TOWARDS MORE AUTONOMOUS AND ADAPTIVE AI SYSTEMS.