ISSN: 2182-2069 / E-ISSN: 2182-2077

Variational Autoencoder Diffusion Model (VAEDM) and Divergence Asynchronous Reinforcement Learning (DARL) for Rail Surface Defect Detection

Samyuktha Sasi Sekaran[1], and Dr.M. Subaji[2*]

[1]Ph.D. Research Scholar (EPT), School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India. samyuktha.ss2015@vit.ac.in, https://orcid.org/0000-0002-5464-1511

[2*]Professor & Director, Institute for Industry and International Programmes, Vellore Institute of Technology, Vellore, India. msubaji@vit.ac.in, https://orcid.org/0000-0001-5332-4656

Received: January 19, 2024; Revised: March 13, 2024; Accepted: April 12, 2024; Published: May 30, 2024

Abstract

The speed and load capacity of trains are improving at an accelerating rate, which raises the standards for railway services' security criteria. The track's surface may progressively reveal varying degrees of defects because of the impacts of moisture, temperature, load, and other factors; if the flaws never addressed in a prompt period, the degree of defects would expand, significantly raising the probability of train service. The development of automatic RSDD (Rail Surface Defect Detection) has substantial practical and scientific implications. In this study, a multi-crack detection technique depends on Variational Autoencoder Diffusion Model (VAEDM), The introduction of VAEDM allows for the non-destructive detection of fastener and RSD. In order to produce hidden elements from images that have various cracks, VAEDM combines the multi-layer and a rail surface image encoder was introduced. Zero Shot- Divergence Asynchronous Reinforcement Learning (ZS-DARL), Policy-Gradient is used to establish the relationship between defects and non-defects. DARL transformer encoder is introduced for taking long-range dependences for FE (Feature Extraction) from detected objects images with various patterns and dimensions. The simulation outcomes on the open Railway Track Fault Detection (RTFD) and Rail Surface Defect Datasets (RSDDs) with rail surface defects are collected from rail tracks surface defect detection. Results are measured using the metrics like recall, precision, F-measure, and accuracy.

Keywords: Rail Surface Defect Detection (RSDD), Variational Autoencoder Diffusion Model (VAEDM), Zero Shot - Divergence Asynchronous Reinforcement Learning (ZS-DARL), and Class Knowledge Graph (CKG).

1 Introduction

Both transporting and regulating the train's speed are track functions in the high-speed rail system. The security of railroad transportation is directly impacted by the way it works. As an outcome, the steel rail needs to be flawless and lack of surface flaws. Unfortunately, surface defects are unavoidable and then developed by deterioration, temperature variations, fatigue loading, and foreign objects among the wheel

Journal of Internet Services and Information Security (JISIS), volume: 14, number: 2 (May), pp. 298-317.
DOI: *10.58346/JISIS.2024.I2.019*

*Corresponding author: Professor & Director, Institute for Industry and International Programmes, Vellore Institute of Technology, Vellore, India.

and rail throughout train operations (Kishore et al., 2019; Shaik et al.,2020), and then spread over repetitive extrusion, that is brought due to the contact tension that exists among the rail and the wheel (Yang et al., 2018). RSD may cause quick degradation and failure that would require expensive repair if they are not found in a timely manner. RSDD depends mostly on observation by railroad operators in a timely manner.

While manual detection offers lower detection efficiency, a higher detection omission rate, and poor performance in real time, it also has the benefit of being easy to use and inexpensive. Conventional detection techniques necessitate manual operation, which is labor-intensive, time-consuming, and inefficient. Additionally, it exposes examiners for predicting risk factors. Subsequent techniques for detection include vision-based techniques (Taştimur et al., 2016) (Zhang et al., 2018), time-frequency analysis, nondestructive evaluation (EC (Eddy Current), UW (Ultrasonic Wave), or AE (Acoustic Emission) (Park et al., 2021), and a combination of the previously mentioned methods (Yu et al., 2018). Because there is insufficient heuristic structure data or texture attributes, the previously mentioned approaches are not very effective for RSDD (Wu et al., 2022). Nevertheless, the signals produced by the flaws in railroad surfaces are extremely weak, making it challenging for the aforementioned techniques for detecting them. Simultaneously, the fault signals are susceptible to interference from the external background, making it challenging to obtain satisfactory outcomes. The technique for RSDD still has a lot of prospective for development. Consequently, there is great practical utility and scientific importance in the automatic RSDD

RSDD employs the MV (Machine Vision) technique, which was made possible by advancements in computer technology. Due to its advantages in terms of speed, accuracy, and dependability, MV is receiving increasing interest from researchers, leading to the development of numerous techniques for the RSD (Trivedi et al,2023; Marangunic et al., 2022). In order to acquire fault information for testing and training models, real detection images are manually screened. Utilizing cameras, conventional ML (Machine Learning)-based detection techniques are employed for RSDD (Jonnerby et al., 2023, Srinivasa et al., 2023; Kim et al., 2019). These techniques demand the creation of manually created or predetermined features by human evaluation of images of RSD, followed by the suggestion of the appropriate FLA (Feature Learning Algorithm) for identification. For RSDD, it must employ the FE approach (Wang et al., 2016) or an operator template and model-based TS (Threshold Segmentation) technique (Banik et al., 2020). Nevertheless, these techniques are vulnerable to defects that might end up in blind spots being detected. This makes achieving strong detection capabilities using MV algorithms is challenging.

The advancement of RSDD technologies has been significantly enhanced by classic ML detection techniques, however these methods fall short in extracting the fault features and have notably poor detection accuracy for small targets. Deep Convolutional Neural Networks (DCNNs) have been established in recent times due to the explosive growth of DL (Deep Learning). These networks are capable of accurately and efficiently FE dynamically. Multiple investigations (Yuan et al., 2019) have successfully classified RSD using deep CNN due to its tremendous capabilities. The inability of these algorithms to identify image imperfections, which is essential in real-world scenarios, is one of their fundamental shortcomings. Moreover, quick inference is not possible with the suggested complex frameworks. Detecting incidents of objects in images is a basic task in CV (Computer Vision), known as Object Detection (OD). It has been extensively employed in numerous sectors throughout the last ten years (Du et al., 2017). In certain domains, the precision of detection has even outperformed human ability (Min et al., 2018). It might be a useful technique for identifying and locating rail flaws. For OD, sophisticated DL-OD methods have been developed. Subsequently, RL (Reinforcement

Learning) evolved rapidly, and because of its superior representation of features and modeling abilities, it has emerged as the distinct solution to RSDD. As an outcome, the study is heavily weighted on the real-time RSDD, which heavily depends on the network's detecting rate.

In this study, a unique OD scheme, and RSDD is introduced to detect rail defects. With the introduction of VAEDM, latent materials can be created from images with various cracks through the combination of a multi-layer with a rail surface image encoder. Zero Shot- Divergence Asynchronous Reinforcement Learning (ZS-DARL), Policy-Gradient is used to establish the relationship between defects and non-defects. ZS-DARL method was tested and compared with other methods. The suggested technique's primary benefits including higher accuracy and lower error. ZS-DARL provides superior accuracy and combined RSDD and localization capabilities.

2 Literature Review

In order to detect road cracks, (Sekar & Perumal, 2021) introduced a new multi-tasking Faster Region CNN (R-CNN) methodology that utilizes the advantage of the Region of Interest (RoI) Align and Global Average Pooling (GAP) approaches. In order to prevent quantizing the stride, ROI Align is employed. For the purpose to map the suggestion to the input image via bi-linear interpolation and minimize data loss. The GAP layer receives the output features from ROI Align and significantly condenses the multi-dimension information into a single FM (Feature Map). The FC (Fully Connected) layer (softmax) and a regression framework are both given the output of the GAP layer in order to estimate the position of the crack with a bounding box. Images from Chennai, Tamil Nadu, India's Outer Ring Road have been collected (a dataset comprising 19300 images). In order to determine the ground-truth label of the bounding boxes for the cracks, the gathered road images were pre-processed utilizing a variety of standard IP (Image Processing) algorithms. The outcomes of classification and detection were assessed utilizing the F1-measure, precision, and recall.

A RSDD technique depends on IE (Image Enhancement) and improved Cascade R-CNN was presented by (Luo et al., 2021). For enhancing the contrast among the defects and the background, the rail surface image is initially processed using the upgraded Retinex technique. Next, an improved Cascade R-CNN is used for RSDD. To address the imbalance among the training and difficult image IoU distributions, the discrepancy among the extracted FM and the RoI due to rounding quantization in the RoI pooling, and the inaccuracy of the regression loss Smooth L1 for the regression of the predicted bounding box, the Intersection over Union (IoU) balanced sampling, RoI align, and complete IoU (CIoU) loss are implemented. At last, in order to address the issue of over-fitting of network training brought through limited image data, the dataset of RSD images is enlarged utilizing techniques including flipping transformation, RC (Random Cropping), brightness transformation, and GANs (Generative Adversarial Networks).

For RSD, (Wang et al., 2022) suggested a novel surface defect detection network based on Mask R-CNN. A new evaluation metric called CIOU is employed in the region proposal network to get around the constraints of IOU in particular situations. In the training stage, both DA (Data Augmentation) and TL (Transfer Learning) are employed to deal with the issue of small defective datasets. The detection network has been developed in a pyramid feature structure for multi-scale fusion. With more accuracy, the fault place can be found using the suggested framework.

A multiobject detection technique based on DCNN was presented by (Zheng et al., 2021, Unisa et al., 2022); it can detect fastener and rail surface flaws non-destructively. Initially the enhanced You Only Look Once v5 (YOLOv5) structure limits the rails and fasteners on the railway image. Next, the RSD are

found and the defect region is divided using the defect detection algorithm according to Mask R-CNN. Ultimately, the fastener state is classified using an algorithm built on the ResNet architecture. Experimental testing is carried out utilizing images of ballast and ballastless railroad tracks that were gathered from the Shijiazhuang-Taiyuan high-speed railroad in order to confirm the reliability and efficacy of the suggested approach.

An ensemble structure for industrialized rail defect detection was created by (Li et al., 2022). To gain features, many backbone networks are utilized separately. These are then combined in a binary format to create sub-networks that are more effective and diverse. To further increase the diversity of the model, random procedures are employed to image augmentation and feature augmentation. To cut down on computing costs and model parameters, an integrated FPN (Feature Pyramid Network) is used. The technique executes better than a single detecting structure in the given rail fault challenge, according to experimental outcomes.

An enhanced YOLO X and IET (Image Enhancement Technique) for RSDD was presented by (Zhang et al., 2023). The surface image of the steel rail is first processed using a fusion IET in the Hue, Saturation, and Value (HSV) space, emphasizing flaws and improving background contrast. Then, for feature fusion in the YOLOX backbone structure, the faster and more effective Bidirectional FPN (BiFPN) is implemented. Furthermore, to improve the capacity to represent image features, the Normalization-based Attention Module (NAM) attention appliance is provided. The technique's RSDD enhances the mean Average Precision (mAP) of the YOLOX networks by 2.42%, according to the testing data.

A Coarse To Fine Model (CTFM) was presented by (Yu et al., 2018) to detect flaws at various scales. Subimage, region, and pixel levels are the 3 scales on which the model operates, ranging from coarse to fine. The baseline subtraction technique utilizes the longitudinal direction’s row consistency to heavily filter the defect-free range at the subimage level, leaving roughly recognized subimages that may include defects. At a higher level, the region extraction approach uses phase-only Fourier transforms (FT) to find specific fault regions. It was influenced by visual saliency models. Pixel consistency is used at the highest stage of the pixel subtraction algorithm to fine-tune each defect's shape. A real rail line and Type-I and Type-II RSDD datasets are used to assess the suggested methodology. Both the defect-level index and the pixel-level index demonstrate that CTFM works better than the most advanced techniques, as demonstrated by the testing data.

The new Rail Border Guidance Network (RBGNet) for salient Rail Surface (RS) detection has been suggested by (Wu et al., 2022). Firstly, a unique design is suggested to precisely identify the RS with well-defined bounds by making full use of the complementarity among the RS and the Rail Edges (RE). Second, to monitor the network and learn the transition among the input and ground truth, a novel hybrid loss comprising of Binary Cross Entropy (BCE), Structural Similarity Index Measure (SSIM), and IoU is presented and prepared into the RBGNet. Lastly, tests carried out on the intricate Unmanned Aerial Vehicle (UAV) rail dataset reveal that the technology is capable of achieving a better DR (Detection Rate) and adapting well to challenging situations.

The pyramid feature CNN for RSDD was introduced by (Liu et al., 2022). Initially, the Pyramid Feature Extraction Module (PFEM) extracts multi-scale FMs according to the attributes of backgrounds and faults. The FMs are then fed into a lightweight network with a limited set of parameters. Utilizing the IOU loss function and the BCE loss function, the network is trained utilizing only forty percent of the dataset. In the study, the RSDD dataset is used to compare the effectiveness of the suggested strategy with alternative approaches.

An improved Unified Perceptual Parsing for Scene Understanding Network (UPerNet) designed specifically for RSDD was presented by (Min et al., 2023). Specifically, the Transformer architecture-based Swin Transformer Tiny version (Swin-T) network is used for proficient FE. This method avoids the problem of inductive preference by utilizing the global data contained in the image. The window-based self-attention, thereby lowering the parameters of the model's count, further increases the effectiveness of the model. For gradient optimization, Cross-GPU Synchronized Batch Normalization (SyncBN) is presented, which incorporates the Lovász-hinge loss function to take advantage of pixel dependence relations.

3 Proposed Methodology

In this study, a multi-crack detection technique depends on VAEDM has been introduced which achieves non-destructive RSD and fastener defects. VAE is a generative framework explicitly designed to capture the underlying probability distribution of a dataset and separate objects from them. Then, Zero Shot-Divergence Asynchronous Reinforcement Learning (ZS-DARL), Policy-Gradient is used to establish the relationship between defects and non-defects. In order to derive FE from OD images, transformer encoders are employed to capture long-range relationships. The suggested technique's general architecture is depicted in Figure 1.

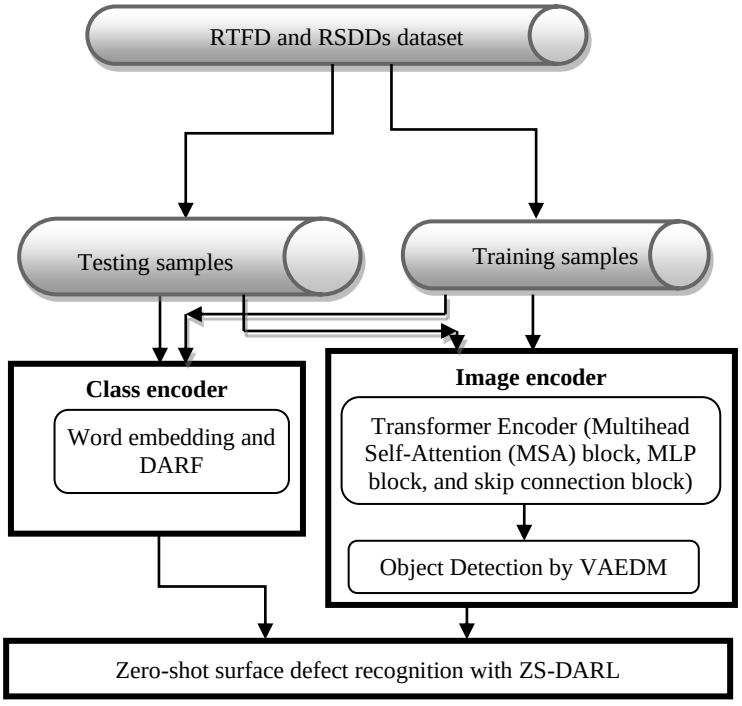

Figure 1: Architecture Diagram of Surface Defect Recognition

1.1. Dataset Description

Two benchmark datasets like RTFD and RSDDs has been introduced for surface defect detection.

RTFD: https://www.kaggle.com/datasets/salmaneunus/railway-track-fault-detection/ data. It consists of defective and non-defective images with training, testing, and validation set with 384 images. Both classes have equal number of images.

RSDDs: Type-I and Type-II rail defect data comprise the RSDDs dataset. 67 defect images taken from high-speed train lines were employed to detect type-I defects. On the other hand, 128 defect images from regular/heavy-duty transportation tracks were used to gather type-II problems. Furthermore, in the training stage, the algorithm requires a finite number of defective images, that will be chosen at random from the test set. Type-I rail images are 160 × 160 in resolution after partitioning and resizing, but Type-II rail images are 64 × 64 in resolution. It has been collected from https://github.com/neu-rail-rsdds/rsdds.

1.2. RSDD

Suggested model includes of IE (Image Encoder), the CE (Class Encoder), and the classifier as illustrated in Figure 2. Initially the IE function encodes an input image as an image feature. Secondly, the CE function encodes classes as the class features. To attain the image label, the classification function receives the image feature and the class value. The training and testing phase comprise the two phases of the proposed architecture as shown in Figure 2.

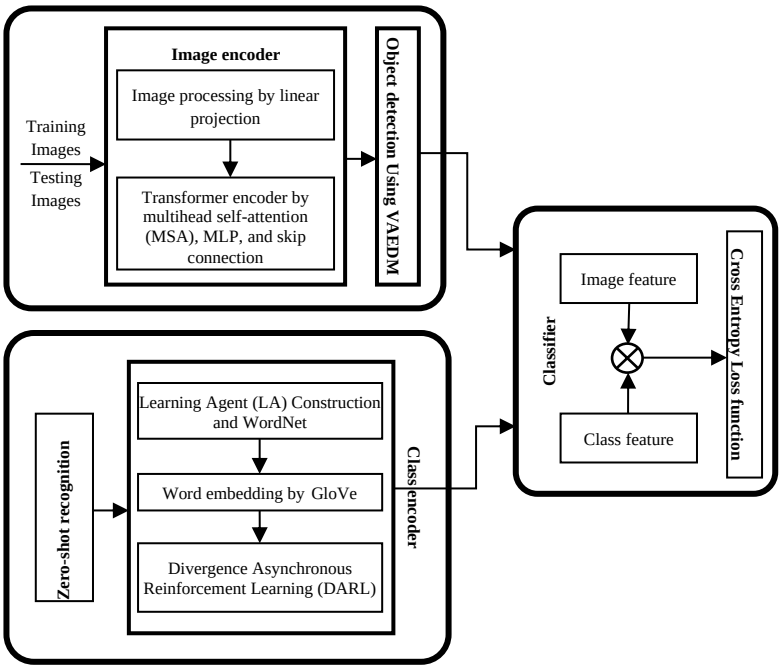

Figure 2: Proposed Multi-crack Detection Model

The IE, which is made up of the TE (Transformer Encoder) and IP, is responsible for extracting the features from the image. The word embedding, the DARL, and the Learning Agent (LA) construction are all included in the CE. It uses the DARL to construct the defect classes in the LA, starts features using word embedding, and then employs the DARL to determine each agent's features. To determine the class score, the classifier module multiply the characteristics of the input image by the class features. Based on this score, the class data is predicted.

Image Encoder

Figure 2 (Kampffmeyer et al., 2019), the image encoder is composed of the TE component and image processing component. The transformer encoder needs the image processed into a certain input form, which is handled by the image processing component. During the training and testing phases, there is

no cross-section amongst the input image classes. To extract individuality, the transformer encoder transmits the previously analyzed image.

Image processing: The image is divided into N fixed-size series patches by the image processing element, and a linear block embeds the input patches into a 1D latent variable. Subsequently, every patch embedding and the specific token have position embeddings, and all patch embeddings have specific tokens added to the front. Thus, the image processing function processes the image to the input format needed through the TE.

TE: Transformer blocks are arranged in L layers within the (TE) Transformer Encoder system. The transformer block consists of the skip connection block, MLP block, Multihead Self Attention (MSA), and normalization (Norm). le_0 is presently employed to denote an input image $x \in \mathbb{R}^{\mathcal{H}\times\mathcal{W}\times\mathcal{C}}$ that is supplied into the transformer encoder element. Firstly, utilizing Norm, MSA, and skip connection blocks, le_0 is denoted as le'_l, and the method is given by equation (1). There, le'_l denotes the l^{th} layer of the transformer block. Secondly, employing MLP, Norm, and skip connection blocks to denote le_l, the procedure may be expressed as equation (2). In image feature, le_0 is ultimately incorporated as le_lover the TE, and [class] token le^0_{tb} is assigned as x' utilizing the Norm block; this procedure may be summed up as equation (3). The transformer encoder module operation stated as follows,

$$le_l^{'} = MSA(Norm(le_{l-1})) + le_{l-1}, \forall l = 1 \cdots tb \quad (1)$$

$$le_l = MLP(Norm(le_l^{'})) + le_l^{'}, \forall l = 1 \cdots tb \quad (2)$$

$$x' = Norm(le^0_{tb}) \quad (3)$$

where tb is the quantity of transformer blocks. The TE phase and the IP phase extract the input image $x \in \mathbb{R}^{\mathcal{H}\times\mathcal{W}\times\mathcal{C}}$ as a 1D vector $x' \in \mathbb{R}^{\mathcal{D}}$.

Object Detection Using VAEDM

A standard normal distribution-following variable is gradually denoising it in order to use the Diffusion Model (DM) to learn a target defect object, which is represented as p(x). As the length of the fixed Markov chain can be denoted as T, the diffusion process can be computed through repeatedly adding noise, represented by $\epsilon \sim N(0, 1)$ to x the initial input image.

$$x_t = \alpha_t x + \beta_t \epsilon \quad (4)$$

Here, the hyperparameters determining the noise region can be represented as α_t and β_t , and t is consistently sampled from 1...T. The following defines the learning objective of common score-matching DM:

$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2] \quad (5)$$

Modeling a CD (Conditional Distribution) p(x|c), where x represents the image representation and c is the condition. Then, equally weighted denoising AE that has been provided to predict that x_{t-1} will be the solution from x_t can be denoted as $\epsilon_\theta(x_t, t)$. A conditional denoising backbone, $\epsilon_\theta(x_t, t, \tau_\theta(c))$, can be trained by DM with the following learning objective, to simulate such CDs.

$$L_{DM} = \mathbb{E}_{x,c,\epsilon\sim\mathcal{N}(0,1),t}[\|\epsilon - \epsilon_\theta(x_t, t, \tau_\theta(c))\|_2^2] \quad (6)$$

Here, the domain-specific encoder that converts c into an intermediate representation is indicated by the symbol τ_θ. With DM, CD p(x|c) is constructed, where the denoising backbone's computational effort is highly dependent on the magnitude of x. Latent DMs (LDMs) conduct diffusion and denoising in a perceptually compressed latent space by using a pre-trained encoder E and decoder D. Since z may be decoded using D to correspond to x, LDM functions inside the latent domain p(z).

The VAE methodology is a revolutionary technique that combines DL and statistical models. The primary differentiator between VAE and conventional AEs is their ability to train latent variables having continuous distributions, which has shown to be an especially helpful feature when addressing computational modeling tasks. Rather than returning discrete values, VAE encoding is wisely built to provide a distribution through the latent space. To be more precise, the E creates a pair of vectors consisting of a standard deviation vector (σ), and a mean vector (μ) and a standard deviation vector (σ). Therefore, unlike typical AEs that learn a deterministic mapping, the VAE aims for learning the distributions of latent features according to the mean values and their variances. The VAE structure is illustrated in Figure 3, and the images of σ and μ values indicate that the latent dimensional space is stochastic. The latent representation coupled with weights can be denoted as Z and biases can be denoted as (ϕ), and X is the encoder model's input.

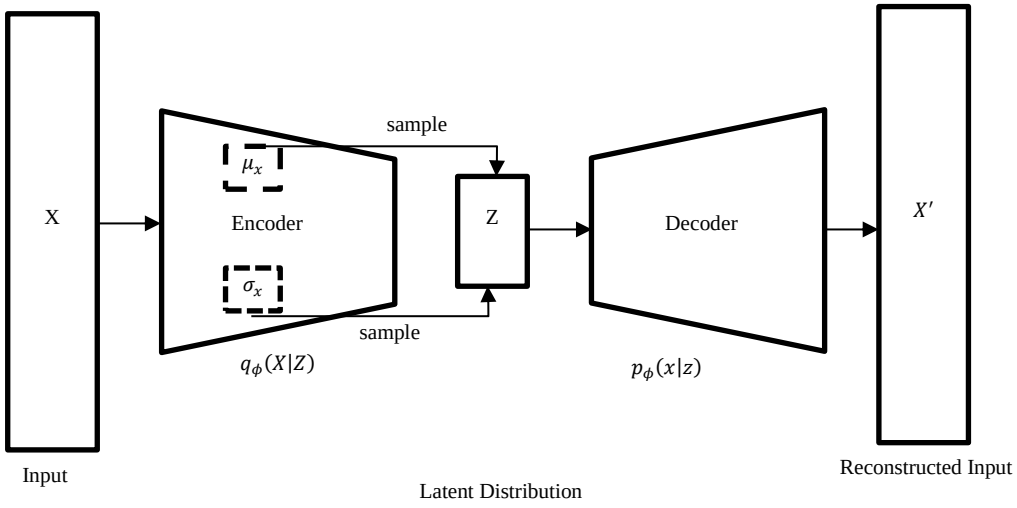

Figure 3: Variational Autoencoder (VAE) Architecture

With x_i standing in for the i^{th} image, label X as the collection of all images in the original dataset. It is encoded to z, or z=$g(X)$, via a function called $g(X)$, whereas the size of z is much smaller than that of X. Once the simplified image dataset z has been provided to the decoder, $\tilde{X}$ is produced by decoding z. Thus, $\tilde{X} = f(z)$ is the mathematical expression for the decoder. The proximity among $\tilde{X}$ and X is then estimated using the loss function $l = \| X - \tilde{X} \|^2$ under an arbitrary norm. The framework is deemed effective if l has a modest magnitude. Assume that in this case, the encoded z will contain the majority of the beneficial information from X, meaning the level once dimensionality reduction is implemented throughout the framework's training phase, z will be sufficient to represent the original dataset.

Let X, for instance, considered an image with the dimensions that contain the data about X are C, H and W. The main objective is to train an AE to reduce the dimensionality of the photo to $z \in \mathbb{R}^d$. After that, a decoder is applied to modify the image to $\tilde{X} \in \mathbb{R}^{C \times H \times W}$ so that the loss function is as little as possible. Since the distribution of z, indicated by $p(z)$, has not been described, in reality this model will produce both undesired noisy constituents and valuable properties of the image. In order to make up for this shortcoming, the VAE was employed to simulate the probabilistic distribution of z before all pertinent information from X was taken out and employed to create a sampling space of z, which was then fed into the decoder to retrieve the image.

Since I is an identity matrix in the case of $z \sim N(0, I)$, z can be considered as a MRV (Multidimensional Random Variable) that follows the conventional MGD (Multivariate Gaussian

Distribution). The related i^{th} images are represented by z_i and x_i, respectively, and we express them as random variables, z and X. Using this configuration, the final result is produced using a two-stage stochastic method where z is considered the hidden variable: (1) the previous distribution of X is encoded and sampled to produce z_i; (2) an image x_i is produced according to the CD $p(X|z_i)$.

In terms of the decoding procedure, the decoder was fed the images z_i that came from the $N(0,\mathbf{I})$ distribution. Next, the parametrized decoder created a mapping that produced an accurate distribution of z_i that corresponded to X, that can be represented as p_ϕ (X|z_i). For each given z_i, X obeys an isotropic MGD, i.e., Equation (7) holds, in order to minimize the statistical complexity. This indicates that the distribution of $X|z_i$ can be derived by fitting μ_i' and $\sigma_i'^2$ after z_i is taken into the decoder.

$$\mathrm{p_\phi(X|z_i) = N\left(X\middle|\mu_i'(z_i;\phi), \sigma_i'^2(z_i;(z_i;\phi)*I\right)} \quad (7)$$

Equation (8) can be derived by considering: $z \sim N(0,\mathbf{I})$, where m is the hyper-parameter in the VAE framework.

$$\mathrm{p_\phi(X) = \frac{1}{m}\sum_{j=1}^{m} p_\phi\left(X\middle|z_j\right)} \quad (8)$$

Next, using the input dataset X as a basis, the Maximum Likelihood Estimation (MLE) is employed to compute ϕ. Equation (9), which displays the detailed formulation, provides

$$\mathrm{\phi^* = argmin_\phi - \sum_{i=1}^{n} \log p_\phi(x_i) = argmin_\phi - \sum_{i=1}^{n} \ln\left(\frac{1}{m}\sum_{j=1}^{m} p_\phi\left(X\middle|z_j\right)\right)} \quad (9)$$

In general, the X dimension is rather huge, whereas the dimension of z is not exceptionally small regardless of the dimensionality reduction method. Thus, in order to obtain a precise computation of $p_\phi(X)$, a sufficiently enough amount of images z_i must be taken into consideration. The PD (Posterior Distribution) $p_\phi(z|x_i)$ must be added to the encoder in order to handle this. The application of the Bayes formula to the computation of $p_\phi(z|x_i)$is demonstrated by equation (10) (Ding, 2022).

$$\mathrm{p_\phi(z|x_i) = \frac{p_\phi(x_i|z)p(z)}{p_\phi(x_i)}} \quad (10)$$

Next, the parametrized encoder and ϕ are optimized using the AutoEconding Variational Bayesian (AEVB) technique (Mak et al., 2023). A PD of the encoder (with parameter θ) is represented by $q_\theta(z|x_i)$. Then the encoder may be used to derive the $z|x_i$ probabilistic distribution (Mak et al., 2023), if $q_\theta(z|x_i) \sim p_\phi(z|x_i)$. $p_\phi(z|x_i)$ is of MGDs since $p(z)$ and $p_\phi(X|z)$ exist. Therefore, all that is needed to describe the posterior of the generative model is to obtain the outputs of σ^2 and μ from the encoder. Equation (11) indicates that for any image x_i, $q_\theta(z|x_i)$ should meet the distribution.

$$\mathrm{q_\theta(z|x_i) = N(z|\mu(x_i;\theta), \sigma^2(x_i;\phi)*I)} \quad (11)$$

Steps 1 through 4 of the VAE algorithm's actual procedure are described below for this research.

Step 1: The data point/image x_i was allocated by the encoder, and NN (Neural Network) methods were used to determine the parameters of $q_\theta(z|x_i)$ that the latent variable z follows. Determining the parameters μ_i and σ_i^2 of the GD (Gaussian distribution) and $z|x_i$ complies with sufficient, as this PD is of an IGD (Isotropic GD).

Step 2: An image, z_i, was extracted from the distribution using the μ_i and σ_i^2 parameters, and it is considered to be x_i of a similar kind.

Step 3: Once $z_i$was allowed into the decoder, the distribution parameters $X|z_i$ might be obtained. This allowed the decoder to determine the likelihood distribution $p_\phi(X|z_i)$ .Indicate the output parameters as μ_i' and $\sigma_i^{2\prime}$ since the likelihood would likewise follow an IGD.

Step 4: Using sampling, a series of data points $\{\tilde{x}_i'\}$ was collected after the statistical parameters of the distribution $X|z_i$ were determined.

$$p_\phi(X|z_i) = N(X|\mu_i'(z_i;\phi), \sigma'^2 * I) \quad (12)$$

Furthermore, equation (12), in which σ'^2 is regarded as a hyper-parameter, can be used to mathematically define $p_\phi(X|z_i)$, which is known to be an isotropic MGD having fixed variance.

Given that $p_\phi(X|z_i)$ is an isotropic MGD with constant variance, it makes sense to define σ'^2 as a K-dimensional vector in this research, where each element has a probability of 0.5. From there, equation (13), the equivalent loss function, can be expressed.

$$L = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(-1 + \sigma_i^{(j)^2} + \mu_i^{(j)^2} - \ln\sigma_i^{(j)^2}) + \frac{1}{n}\sum_{i=1}^{n}\left\|x_i - \mu_i'^2\right\| \quad (13)$$

Here, z_i is imaged from $z|x_i$ and functions as the input of the decoder; μ_i's is the output of the decoder, that particularly corresponds to the ultimately generated data point $\tilde{x}_i$; and x_i symbolizes the i^{th} image, which serves as the input of the encoder; μ_i and σ_i^2 are the outputs of the encoder, which serve as the parameters of the distribution of $z|x_i$.

Class Encoder

Class encoder system includes the Learning Agent (LA) Construction, word embedding, and DARL system. The defect LA is generated utilizing the surface defect class data provided by the DARL function. The agents in the generated DARL have their features prepared by the word embedding function. When agents and the defect LA are supplied into the DARL encoder part it may acquire every agent visual representation.

Zero-shot recognition (ZSR): Learning a model which is generalize to novel classes that doesn't appear in the training phase is the objective of ZSR. According to formal definitions, a dataset X contains samples for a training set $X_{train} = \{(x_{train}^1, y_{train}^1), \cdots, (x_{train}^n, y_{train}^n)\}$ and a test set $X_{test} = \{(x_{test}^1, y_{test}^1), \cdots, (x_{test}^n, y_{test}^n)\}$, where the base classes are represented as $y_{train}^n \in Y_{base}$, novel classes are represented as $y_{test}^n \in Y_{novel}$, and $Y_{base} \cup Y_{novel} = Y$. A fully disjoint collection of classes is employed to train and test the zero-shot framework: The ability of $Y_{base} \cap Y_{novel} = \emptyset$ to transmit data from Y_{base} to Y_{novel} is crucial.

Since the training and testing data are entirely distinct sets of labels and non-i.i.d. It is usually challenging to generalize a direct mapping $f(\cdot) : X \to Y$ from training data for labeling in ZSR situations. Therefore, in order to serve as a link among base classes and novel classes, the latent semantic space $\boldsymbol{S}$ for every class becomes available throughout both training and testing. More specifically, $s_y \in S$ the semantic embedding vector is linked to each class $y \in \boldsymbol{Y}$. A solution would acquire by mapping from data to latent semantic space, or $f(\cdot) : \boldsymbol{X} \to \boldsymbol{S}$, and then apply it to the testing data. This approach is mostly dependent on the mapping's capacity to transfer information from $Y_{base}$to Y_{novel}. Therefore, learning a mapping from data to latent semantic space $f(\cdot) : \boldsymbol{X} \to \boldsymbol{S}$ and generalizing on the testing data is an alternative.

Learning Agent (LA) Construction: RL is a ML method for developing satisfactory policies to address class encoder problem (Arulkumaran et al., 2017). A typical RL model is characterised as follows: S stands for the state space, A for the action space, P for the state transition probability, and R for the reward function in a 4-tuple (S, A, P, R).LA learns to perform in a given state to maximise future rewards through interaction with the environment. It is important to note that when making decisions, it is important to balance the agent's short-term and long-term gains (François-Lavet et al., 2018). RL agent is capable of picking the appropriate action for each state with the highest cumulative rewards based on cumulative learning experience.

Word embedding: LA parameters are to be initialized in this part. Every agent in the LA has its attribute generated utilizing word embeddings that were pre-trained on extensive language datasets. Pre-trained on Wikipedia 2014 and Gigaword 5, GloVe is employed by the ZS- LA to map every class into a 300-D vector form. An unsupervised learning system called GloVe is employed to generate vector illustrations for words. After training on consolidated global word-word combination statistics from a corpus, intriguing linear components of the word vector space are displayed in the final visualizations.

DARL MODEL: LA maps inputs to state data in RL by sensing the environments in discrete time periods. LA takes action and monitors the background's response, which takes the shape of incentives or penalties. Agents notice alterations to the environments and adjust their policies to maximize the possible benefit after acting based on states and earning a reward (Figure 4 represents the communication between the agent and environment). Determining the best course of action which optimizes the cumulative benefits is the objective of the optimal policy.

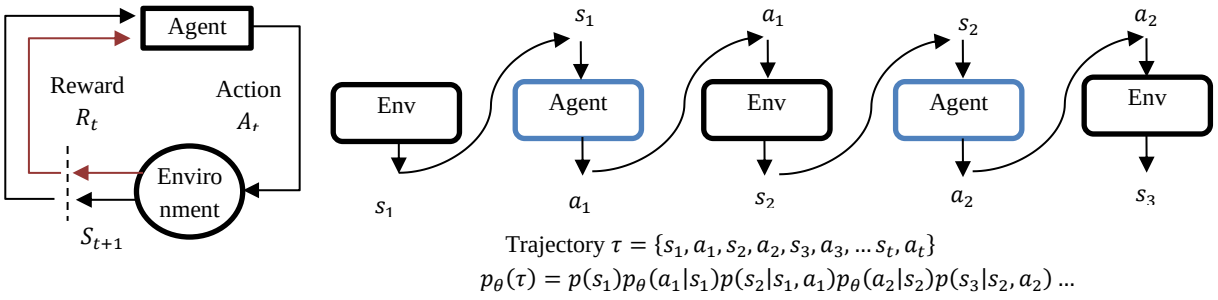

Figure 4: Agent Environment Interaction (Left), a and r Sequences (Right)

Agent behavior is defined by policies, which may be thought of as a mapping function between states to actions. The likelihood that agents will act in a certain way under a certain state's is represented by the policy's model, p(a|s). NNs allow for the realization of policies since S and A represent the input and output, correspondingly. Prospective views were performed with the NN map created with states/observations and actions. The predicted return between state's to policy π (whereG_t is the overall R between S to the conclusion of the episode and γ is the decaying factor) is the state-value function of a Markov Decision Process (MDP).

$$\upsilon_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma \upsilon_\pi(s')] \quad (14)$$

With a policy π and an MDP M = (S,A,P,R,γ), the state series $S_1, S_2, .. S_n$ is a Markov process (S, p^π) with $S_1, R_2, S_2, R_2, \ldots., S_n, R_n$, as the state reward sequence. $(S, p^\pi, R^\pi, \gamma)$ is an MRP.

$$\upsilon_\pi(s) = \sum_{a \in A} \pi(a|s) \left\{ r(s, a) + \gamma \sum_{s' \in S} T(s'|s, a) \upsilon_\pi(s') \right\} \quad (15)$$

Here, the state transition matrix between state's to s' with policy π can be represented as $T^{\pi}_{s,s'}$, and reward at state's of the agent π can be denoted as r^{s}_{π}. It is possible to express the value function in matrix form.

$$\vec{\upsilon}_{\pi} = \mathrm{r}^{\pi} + \gamma T^{\pi}\vec{\upsilon}_{\pi} \quad (16)$$

Here, the transition matrix can be denoted as T π and a vector ($\vec{\upsilon}_{\pi}$). The matrix is solved using the iterative approach. The expected return at beginning from state s, acting on a, and then implementing policy π is represented by the State-action value function $q_{\pi}(s,a)$ ((Deep Reinforcement Learning and Control Lectures, 2021)).

$$q_{\pi}(s,a) = \mathrm{E}_{\pi}[\mathrm{R}_{t+1} + \gamma q_{\pi} S_{t+1}, A_{t+1} | \mathrm{S}_{\mathrm{t}} = \mathrm{s}, \mathrm{A}_{\mathrm{t}} = \mathrm{a}] \quad (17)$$

The calculation of $\upsilon_{\pi}(\mathrm{s})$ and $q_{\pi}(s,a)$ can be seen in Figure 5. In procedure, the prediction is frequently expressed in the form of random reward (Deep Reinforcement Learning and Control Lectures, 2021).

$$q_{\pi}(s,a) = \sum_{s',\mathrm{r}} \mathrm{p}(\mathrm{s}',\mathrm{r}|\mathrm{s},\mathrm{a}) \left[\mathrm{r} + \gamma \sum_{a'} \pi(\mathrm{a}|\mathrm{s}) \mathrm{q}_{\pi}(\mathrm{s}',\mathrm{a}')\right] \quad (18)$$

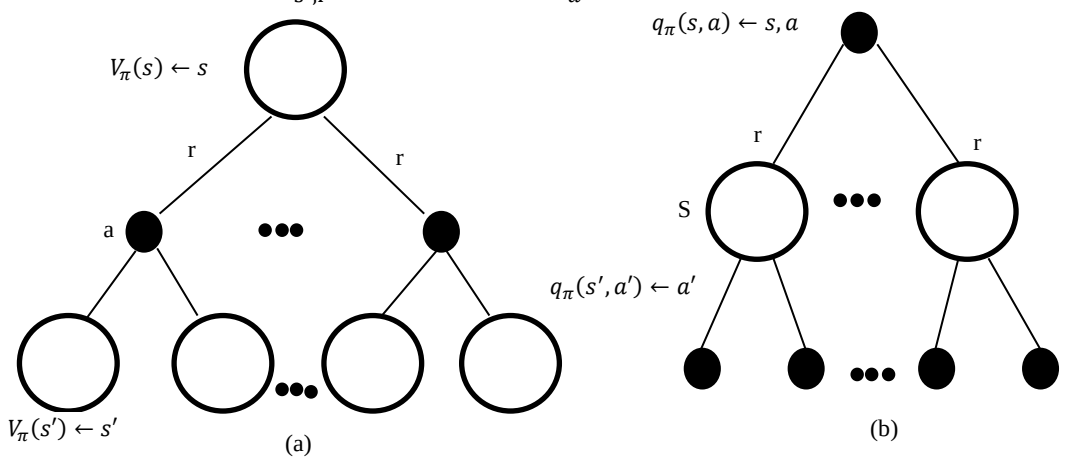

Figure 5: (a) Illustration of State Value Computation. (b) Action Value

By utilizing SG (Stochastic Gradient), Policy Gradient (PG) is applied for minimizing policy loss and is represented in the following way (Deep Reinforcement Learning and Control Lectures, 2021),

$$\nabla\overline{\mathrm{R}_{\theta}} = \frac{1}{\mathrm{N}} \sum_{\mathrm{n}=1}^{\mathrm{N}} \sum_{\mathrm{t}=1}^{\mathrm{T_n}} \left(\sum_{\mathrm{t}'=\mathrm{t}}^{\mathrm{T_n}} \gamma^{\mathrm{t}'-\mathrm{t}} \mathrm{r}_{\mathrm{t}'}^{\mathrm{n}} - \mathrm{b} \right) \nabla \log \mathrm{P}_{\theta}\left(\mathrm{a}_{\mathrm{t}}^{\mathrm{n}} | \mathrm{s}_{\mathrm{t}}^{\mathrm{n}}\right) \quad (19)$$

Here, $V^{\pi}(s_t^n)$ is represented by the baseline, the time decaying factor can be symbolized as γ, and the reward between time step t' to the episode's end can be denoted as $r_{t'}^{n}$. The absence of expectation E, $Q^{\pi}(s_t^n, a_t^n) = E[r_t^n + V^{\pi}(s_{t+1}^n)]$, introduces unexpectability. The gradient is approximated as follows ((Deep Reinforcement Learning and Control Lectures, 2021)),

$$\nabla\overline{\mathrm{R}_{\theta}} = \frac{1}{\mathrm{N}} \sum_{\mathrm{n}=1}^{\mathrm{N}} \sum_{\mathrm{t}=1}^{\mathrm{T_n}} \left(\sum_{\mathrm{t}'=\mathrm{t}}^{\mathrm{T_n}} \mathrm{r}_{\mathrm{t}}^{\mathrm{n}} + \mathrm{V}^{\pi}\left(\mathrm{s}_{\mathrm{t+1}}^{\mathrm{n}}\right) - \mathrm{V}^{\pi}\left(\mathrm{s}_{\mathrm{t}}^{\mathrm{n}}\right) \right) \nabla \log \mathrm{P}_{\theta}\left(\mathrm{a}_{\mathrm{t}}^{\mathrm{n}} | \mathrm{s}_{\mathrm{t}}^{\mathrm{n}}\right) \quad (20)$$

Double Deep Q Network (DDQN): to get at the predictable q value $r_t + Q^{\pi}(s_{t+1}, \pi(s_{t+1}))$, the target network's parameters are first copied from the present network of Q. Utilizing SGD, the training

images have been created to minimize the difference among $Q^{\pi}(s_t, \pi(s_t))$& $r_t + Q^{\pi}(s_{t+1}, \pi(s_{t+1}))$. The state value function (critic) and the DDQN (actor) are combined in actor-critical network topologies. In Q -learning, the critic fails to select the course of action; rather, it assesses the actor's quality using the total rewards earned from visiting states. The value of the state is represented by $V^{\pi}(S_a)$ the output scalar, when the state is fed to V^{π} actor. Equation (21) provides an expression for the PGD (Deep Reinforcement Learning and Control Lectures, 2021).

$$J_{PPO}^{\theta'} = J^{\theta'}(\theta) - \beta KL(\theta, \theta') \quad (21)$$

$$J^{\theta'}(\theta) = E_{(s_t, a_t)\sim\pi_{\theta'}}\left[\frac{p_{\theta}(a_t|s_t)}{p_{\theta'}(a_t|s_t)} A^{\theta'}(s_t, a_t)\right] \quad (22)$$

Here, β is employed as the regularizer to manage the variance and the behavior change is represented by the KL divergence of θ, θ'. Asynchronous computing has multiple independent workers, each with a unique weight, interacting in parallel with an exact copy of the environment. The employees receive training concurrently, and they update the global at a "asynchronous" time interval that contains shared parameters on a regular basis. The global changes it after receiving a worker's update, and the worker updates it afterwards. After every update, the workers synchronize with the newly updated global and adjust their parameters with it.

When every worker adjusts weights in accordance with the global, parameter data passes among workers and from workers to the global. There are 5 stages in the cycle process of training. At first, (1) every worker reset to the global network. (2) They then interact with their own environments, (3) determine values and policy losses, (4) determine gradients from losses, average their gradients across all workers for updating the global NN weights, and (5) repeat steps 1 through 5 until convergence or the time maximum is attained. Neural net images are created for every worker based on the batches in the database, ($s_{t_1} a_{t_1} s_{t_1+1}, s_{t_i} a_{t_i} s_{t_{i+1}} \ldots$). Then (state value NN ($\theta_{\upsilon'}$ for worker, $\theta_{\upsilon}$for global). and (parameter $\theta'$for worker, θ for global) are the states that are then inputted into the policy NN

The NN approximator of the policy function is trained using the variance among the estimated (VF)Value Function R_t and the real VF V_t (advantage value $A_t = R_t - V_t$). If a positive advantage is obtained, raise the likelihood of action a_t^n; if a negative advantage is obtained, lower that probability. Then, each worker submits its updated θ' and $\theta_{\upsilon'}$parameters to the global. To avoid miscalculating certain steps, the derivative to the global parameter $d\theta, d\theta_{\upsilon}$ is determined on the average of every work θ', $\theta_{\upsilon'}$. To promote exploration and avoid early suboptimal convergence, the entropy of policy $(H\left(\pi(a_j|s_j;\theta')\right)$ is included. The following is the formalization of the Value function in RL, or V_{π}, performance:

$$V_{\pi} = \sum_{s\in S} w^{\pi}(s) \sum_{a\in A} R(s,a)\pi(s,a) \quad (23)$$

Here, the cumulative reward beginning from state s and action a can be represented as R(s,a), and the likelihood of the entire structure in the states can be denoted as $w^{\pi}(s)$. The expected cumulative reward after π is the performance. The network that received the action and current state as inputs and delivered the state value from the episode as the whole was identified as V_{π}. The most effective performances π^*, denoted by $\pi^* = argmax_{\pi} V_{\pi}$, are satisfied by the optimal performance. Constant improvement of π results from experience-based learning.

Classifier Using DARL System

To determine the class score, the classifier module combines the characteristics of the input image by the class features. Based on this score, the class data is predicted. The image feature obtained from the IE system was multiplied by the class feature supplied using the CE system by the classifier system. The TE module extracts batch images using $x' \in \mathbb{R}^{n \times D}$ image features, while n is the number of batch photos. Class features $p \in \mathbb{R}^{t \times Q}$, and t is the total amount of classes, are obtained by the CE module.

Initially, 2 linear blocks are employed to map the class features and image features to similar size C. Next, to obtain the score $z \in \mathbb{R}^{n \times t}$ for each image that belongs to a class, the classifier module combines the image features and the class features using a linear transformation. Ultimately, the labeling of the batch photos is obtained by using the argmax score. The classifier's operation can be mathematically stated as follows:

$$z = \mathrm{f}\left(\mathrm{x}', \mathrm{W}_{\mathrm{x}'}, \mathrm{W}_{\mathrm{p}}, \mathrm{p}\right) = \left(\mathrm{x}'\mathrm{W}_{\mathrm{x}'}\right)\left(\mathrm{pW}_{\mathrm{p}}\right)^{\mathrm{T}} \quad (24)$$

Here, two trainable weight matrices of linear block are $\mathrm{W}_{\mathrm{p}} \in \mathbb{R}^{\mathrm{F} \times \mathrm{C}}$ and $\mathrm{W}_{\mathrm{x}'} \in \mathbb{R}^{\mathrm{D} \times \mathrm{C}}$. The ZS-DARL algorithm determines the loss and BP (Back Propagation) throughout the training period. The conventional CE loss is employed through the loss function,

$$L(y, z) = -\sum_{\mathrm{i}=1}^{\mathrm{t}} \mathrm{y_i} \log \mathrm{p_i} = -\log \frac{\mathrm{e}^{\mathrm{z_y}}}{\sum_{y_i \in [t]} \mathrm{e}^{\mathrm{z_{y_i}}}} \quad (25)$$

Here, the total amount of classes can be denoted as t, the image's class label is y, the class score is $\mathrm{z_y}$, and $\mathrm{z_{y_i}}$ is the class score. In order to maximize the ZS-DARL framework minimize equation (25) instead. As usual, $argmax_{y \in [t]}\, z$ is expected to occur during the testing period.

4 Results and Discussion

A single NVIDIA RTX2080S GPU, 16GB of RAM, Intel Core i7-9700 CPU, Windows 10 OS, and the YOLO technology are all included in the test environment. The simulation involves adjusting the image size to 224 × 224 pixels, employing the Adam optimization technique to improve the framework, setting the learning rate to 0.0001, training the framework over 100 epochs utilizing a mini-batch of 10. In this case, consider 30% of the information as the test set and 70% of the information as the train set from Rail-5k and RSDDs. The study chooses precision, recall, F-measure, accuracy, and additional variables to contrast the algorithm with in order to appropriately assess its impact.

Equation (26) displays the percentage of actual positive images amongst those determined as positive images, defines precision.

$$\text{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (26)$$

Equation (27) defines recall, shows the percentage of positive instance in the image are properly recognized.

$$\text{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (27)$$

It is insufficient to evaluate the model accuracy solely on recall or precision. As a result, the F-measure was created to take recall and precision into account simultaneously. Equation (28), which defines the F-measure, is an example.

$$F - Measure = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \quad (28)$$

Typically, accuracy is employed to assess an algorithm global accuracy, that can't include too much data or provide a complete assessment of the model. Equation (29) provides the explanation,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (29)$$

True Positive (TP) denoted a properly determined positive instance; True Negative (TN) represented a precisely discovered negative instance; False Positive (FP) represented a misidentified negative instance; and False Negative (FN) represented a mistakenly discovered positive instance for a misidentified negative instance. Accuracy, recall, precision, and f-measure for RTFD and RSDDs among defect detection methods are displayed in Figures 6-9. CTFM (Yu et al., 2018), Deep Convolutional Neural Network (DCNN) (Liang et al., 2018), R-CNN (Sekar & Perumal, 2021), Zero Shot-Semi Supervised Fuzzy Class Knowledge Graph (ZS-SSFCKG), and proposed system with their evaluation metrics are discussed in Table 1.

Table 1: Comparative Results Analysis for Datasets

DATASETS	METRICS	CTFM	DCNN	R-CNN	ZS-SSFCKG	ZS-DARL
RTFD	**PRECISION (%)**	78.55	80.42	82.19	84.74	90.22
	RECALL (%)	79.84	82.15	85.47	87.15	92.52
	F-MEASURE (%)	79.19	81.28	83.79	85.93	91.35
	ACCURACY (%)	78.51	80.86	82.33	84.87	92.47
RSDDs	**PRECISION (%)**	76.18	78.53	82.47	85.25	91.71
	RECALL (%)	78.37	80.31	84.66	86.58	93.15
	F-MEASURE (%)	77.26	79.41	83.55	85.90	92.42
	ACCURACY (%)	79.85	82.59	85.82	87.61	93.88

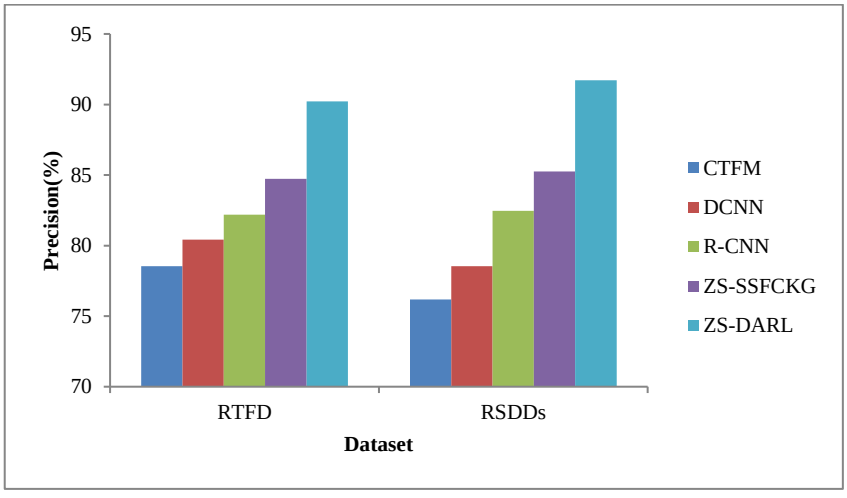

Figure 6: Precision vs. Zs based on Kg Techniques

Figure 6 shows the precision comparison of methods with 2 datasets. From the graph, it can be observed that the proposed system produces superior surface defect identification of 91.71%, previous approaches is 76.18%, 78.53%, 82.47% and 85.25 for RSDDs. Other methods like CTFM, DCNN, R-CNN and ZS-SSFCKG had lesser results of 15.53%, 13.18%, 9.24%, and 6.46% for RSDDs.

Proposed system utilizes the RL, and VAE, making it a superior option for the prompt detection of surface defects.

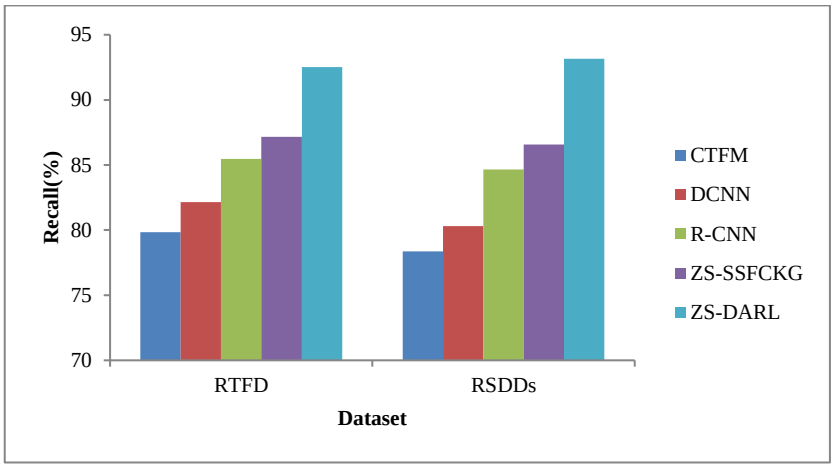

Figure 7: Recall vs. ZS based on Kg Techniques

Figure 7, this graph clarifies the recall comparison of methods with 2 datasets. From the graph, it can be observed that the proposed system produces superior surface defect identification is 93.15%, the previous approaches is 78.37%, 80.31%, 84.66% and 86.58% for RSDDs. Other methods like CTFM, DCNN, R-CNN and ZS-SSFCKG had lesser results of 14.78%, 12.84%, 8.49%, and 6.57% for RSDDs. CTFM, DCNN, R-CNN and ZS-SSFCKG had lesser results of 79.84%, 82.15%, 85.47% and 87.15% for RTFD. From the railway surface defect dataset, the proposed system has the largest recall due of optimal detection of objects from the railway images using the VAE, and then it has been correctly classified using LA.

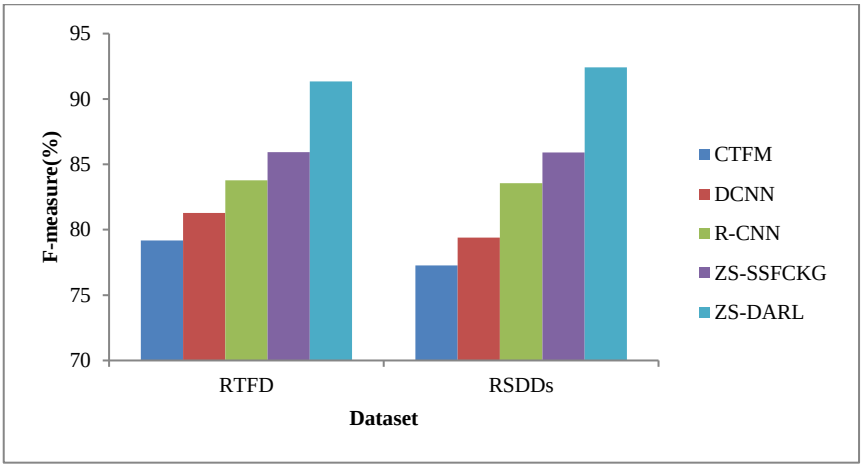

Figure 8: F-measure vs. ZS based on Kg Techniques

Figure 8, this graph clarifies the F-measure comparison of methods with 2 datasets. It shows that the proposed system produces best results of 92.42%, existing methods have given lowest results of 76.08%, 77.26%, 79.41%, 83.55% and 85.90% for RSDDs. Other methods like CTFM, DCNN, R-CNN and ZS-SSFCKG had lesser results of 15.16%, 13.01%, 8.87%, and 6.52% for RSDDs. CTFM, DCNN, R-CNN and ZS-SSFCKG have lesser results of 79.19%, 81.28%, 83.79% and 85.93% for RTFD. It is found that the results of the proposed system provide a Divergence Asynchronous Reinforcement

Learning (DARL) with reduced overfitting and consistently improving learning from the experience than the existing methods.

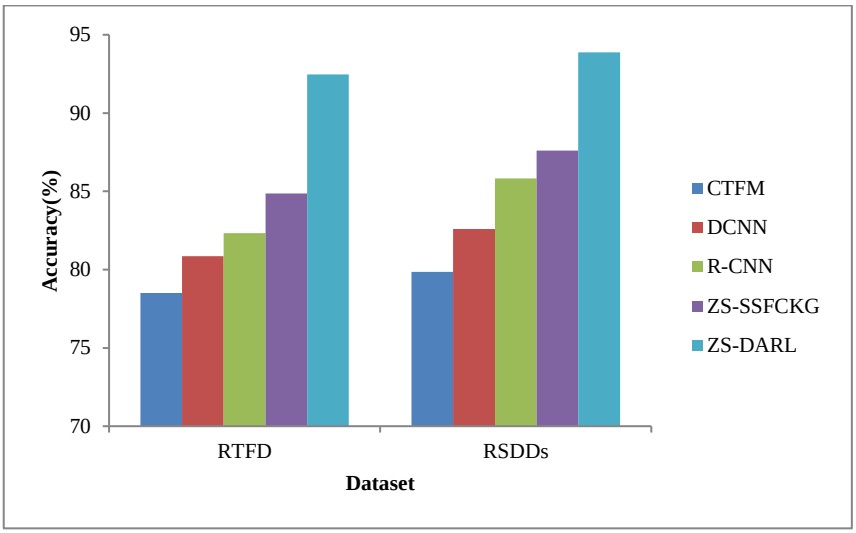

Figure 9: Accuracy vs. ZS based on Kg Techniques

Figure 9, this graph clarifies the accuracy comparison of methods with 2 datasets. Proposed system produces superior results of 92.47%, previous approaches have given results of 79.85%, 82.59%, 85.82% and 87.61% for RSDDs. Other methods like CTFM, DCNN, R-CNN and ZS-SSFCKG had lesser results of 14.03%, 11.29%, 8.06%, and 6.27% for RSDDs. CTFM, DCNN, R-CNN and ZS-SSFCKG have lesser results of 78.51%, 80.86%, 82.33% and 84.87% for RTFD. Proposed system was developed to increase detection rate due to object detection of rail track cracks via VAM, and LA quickly detects the cracks which led the detection accuracy gets improved than the other methods.

5 Conclusion and Future Work

RSDD is crucial to make certain the smooth, secure, and quick function of trains. In this paper, Variational Autoencoder Diffusion Model (VAEDM) is introduced for multi-crack object detection. VAEDM is designed to learn target defects by progressively denoising an image with normal distribution. VAEDM model is introduced for learning the distributions of latent features depends on the mean values and their variances. VAEDM is worked based on the stochastic manner. ZS-DARL framework, class encoder system includes the Learning Agent (LA) Construction, word embedding, and DARL system. The defect LA is generated utilizing the surface defect class data provided by the DARL function. The agents in the generated DARL have their features prepared by the word embedding function. When agent properties and the defect LA are supplied into the DARL encoder part it may acquire every agent visual representation. The asynchronous DARL systems consist of multiple independent workers, each having a weight and interacting together with an exact copy of the environment. The workers receive asynchronous training in parallel and update asynchronously on a regular basis according to their pace of learning for crack detection. Extensive testing on reference datasets like RTFD and RSDDs demonstrates the efficacy of the suggested methodology. F-measure, accuracy, precision, and recall are employed for estimating the performance of RSDD techniques. Subsequent investigations will examine the identification of RSD in intricate settings and enhance the network model's lightweight design while preserving detection precision. Utilize your deeper

understanding of DL in the future to enhance the RSDD algorithm by taking defect segmentation under consideration.

References

[1] Arulkumaran, K., Deisenroth, M.P., Brundage, M., & Bharath, A.A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, *34*(6), 26-38.

[2] Banik, P. P., Saha, R., & Kim, K.D. (2020). An automatic nucleus segmentation and CNN model based classification method of white blood cell. *Expert Systems with Applications*, *149*, 113211. https://doi.org/10.1016/j.eswa.2020.113211

[3] Deep Reinforcement Learning and Control Lectures. (2021). http://www.andrew.cmu.edu/course/10-403/

[4] Ding, M. (2022). The road from MLE to EM to VAE: A brief tutorial. *AI Open*, *3*, 29-34.

[5] Du, X., Dai, P., & Li, Y. (2017). Automatic detection algorithm of railway plug based on deep learning. *Chin. J. of the China Railw. Soc*, *38*(3), 89-96.

[6] François-Lavet, V., Henderson, P., Islam, R., Bellemare, M.G., & Pineau, J. (2018). An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, *11*(3-4), 219-354.

[7] Jonnerby, J., Brezger, A., & Wang, H. (2023). Machine learning based novel architecture implementation for image processing mechanism. *International Journal of Communication and Computer Technologies (IJCCTS), 11*(1), 1-9

[8] Kampffmeyer, M., Chen, Y., Liang, X., Wang, H., Zhang, Y., & Xing, E.P. (2019). Rethinking knowledge graph propagation for zero-shot learning. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11487-11496.

[9] Kim, K. C., Ko, E., Kim, J., & Yi, J. H. (2019). Intelligent Malware Detection Based on Hybrid Learning of API and ACG on Android. *Journal of Internet Services and Information Security, 9*(4), 39-48.

[10] Kishore, M.B., Park, J.W., Song, S.J., Kim, H.J., & Kwon, S.G. (2019). Characterization of defects on rail surface using eddy current technique. *Journal of Mechanical Science and Technology*, *33*, 4209-4215.

[11] Li, H., Wang, F., Liu, J., Song, H., Hou, Z., & Dai, P. (2022). Ensemble model for rail surface defects detection. *PLoS one*, *17*(5), 1-17.

[12] Liang, Z., Zhang, H., Liu, L., He, Z., & Zheng, K. (2018). Defect detection of rail surface with deep convolutional neural networks. *In IEEE 13th World Congress on Intelligent Control and Automation (WCICA)*, 1317-1322.

[13] Liu, Y., Xiao, H., Xu, J., & Zhao, J. (2022). A rail surface defect detection method based on pyramid feature and lightweight convolutional neural network. *IEEE Transactions on Instrumentation and Measurement*, *71*, 1-10.

[14] Luo, H., Li, J., & Jia, C. (2021). Rail surface defect detection based on image enhancement and improved cascade R-CNN. *Laser & Optoelectronics Progress*, *58*(22), 324-335.

[15] Mak, H.W.L., Han, R., & Yin, H.H. (2023). Application of variational auto Encoder (VAE) model and image processing approaches in game design. *Sensors*, *23*(7), 3457. https://doi.org/10.3390/s23073457

[16] Marangunic, C., Cid, F., Rivera, A., & Uribe, J. (2022). Machine Learning Dependent Arithmetic Module Realization for High-Speed Computing. *Journal of VLSI Circuits and Systems, 4*(1), 42-51.

[17] Min, Y., Li, J., & Li, Y. (2023). Rail Surface Defect Detection Based on Improved UPerNet and Connected Component Analysis. *Computers, Materials & Continua*, *77*(1), 941-962.

[18] Min, Y., Xiao, B., Dang, J., Yue, B., & Cheng, T. (2018). Real time detection system for rail surface defects based on machine vision. *EURASIP Journal on Image and Video Processing*, *2018*(1), 1-11.

[19] Park, J.W., Lee, T.G., Back, I.C., Park, S.J., Seo, J.M., Choi, W.J., & Kwon, S.G. (2021). Rail surface defect detection and analysis using multi-channel eddy current method-based algorithm for defect evaluation. *Journal of Nondestructive Evaluation*, *40*, 1-12.

[20] Sekar, A., & Perumal, V. (2021). Automatic road crack detection and classification using multi-tasking faster RCNN. *Journal of Intelligent & Fuzzy Systems*, *41*(6), 6615-6628.

[21] Shaik, S. (2020). A coplanar wave guide fed compact antenna for navigational applications. *National Journal of Antennas and Propagation (NJAP), 2*(1), 7-12.

[22] Srinivasa Rao, M., Praveen Kumar, S., & Srinivasa Rao, K. (2023). Classification of Medical Plants Based on Hybridization of Machine Learning Algorithms. *Indian Journal of Information Sources and Services, 13*(2), 14–21.

[23] Taştimur, C., Karaköse, M., Akın, E., & Aydın, İ. (2016). Rail defect detection with real time image processing technique. *In IEEE 14th international conference on industrial informatics (INDIN)*, 411-415.

[24] Trivedi, J., Devi, M. S., & Solanki, B. (2023). Step Towards Intelligent Transportation System with Vehicle Classification and Recognition Using Speeded-up Robust Features. *Arhiv za tehničke nauke, 1*(28), 39-56.

[25] Unisa, S.A., & Gowda, K.J. (2022). Estimating CSI for Future Generation MIMO Networks Using Deep Learning Techniques and its Applicability to Varied Environments. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 13*(4), 124-136.

[26] Wang, F., Xu, T., Tang, T., Zhou, M., & Wang, H. (2016). Bilevel feature extraction-based text mining for fault diagnosis of railway systems. *IEEE transactions on intelligent transportation systems*, *18*(1), 49-58.

[27] Wang, H., Li, M., & Wan, Z. (2022). Rail surface defect detection based on improved Mask R-CNN. *Computers and Electrical Engineering*, *102*, 108269. https://doi.org/10.1016/j.compeleceng.2022.108269

[28] Wu, Y., Qin, Y., Qian, Y., Guo, F., Wang, Z., & Jia, L. (2022). Hybrid deep learning architecture for rail surface segmentation and surface defect detection. *Computer-Aided Civil and Infrastructure Engineering*, *37*(2), 227-244.

[29] Yang, R., Cao, S., Kang, W., Li, J., & Jiang, X. (2018). Mechanism analysis of spalling defect on rail surface under rolling contact conditions. *Mathematical Problems in Engineering*, *2018*, 1-10.

[30] Yu, H., Li, Q., Tan, Y., Gan, J., Wang, J., Geng, Y.A., & Jia, L. (2018). A coarse-to-fine model for rail surface defect detection. *IEEE Transactions on Instrumentation and Measurement*, *68*(3), 656-666.

[31] Yuan, H., Chen, H., Liu, S., Lin, J., & Luo, X. (2019). A deep convolutional neural network for detection of rail surface defect. *In IEEE vehicle power and propulsion conference (VPPC)*, 1-4.

[32] Zhang, C., Xu, D., Zhang, L., & Deng, W. (2023). Rail Surface Defect Detection Based on Image Enhancement and Improved YOLOX. *Electronics*, *12*(12), 2672. https://doi.org/10.3390/electronics12122672

[33] Zhang, H., Jin, X., Wu, Q.J., Wang, Y., He, Z., & Yang, Y. (2018). Automatic visual detection system of railway surface defects with curvature filter and improved Gaussian mixture model. *IEEE Transactions on Instrumentation and Measurement*, *67*(7), 1593-1608.

[34] Zheng, D., Li, L., Zheng, S., Chai, X., Zhao, S., Tong, Q., & Guo, L. (2021). A defect detection method for rail surface and fasteners based on deep convolutional neural network. *Computational intelligence and neuroscience, 2021*, 1-15.

Authors Biography

Samyuktha Sasi Sekaran, with 11 years' experience in ML, AI, JavaScript, Node.js, and DBMS, specializes in ML/AI for 7 years. Currently pursuing a Ph.D. at VIT University, India, she's a Data Scientist at SAP India. Holding Bachelor's and Master's degrees in Engineering, she's proficient in Python, PyTorch, Keras, OpenAI, and Generative AI, as well as cloud platforms like Kubernetes, GCP, and Azure. Skilled in JavaScript, Node.js, and React, she specializes in Genetic Algorithms, NLP, and Computer Vision, with expertise in Big Data tech like Apache Spark, SQL, and NoSQL databases. She conducted training sessions at SAP on deep learning and ML techniques, excelling in designing and developing commercial applications, contributing significantly to various sectors, recognized for her adaptability, communication skills, and proactive approach.

Dr.M. Subaji obtained his Bachelor’s degree in Mechanical Engineering, Master’s degree in Business Administration and PhD from Kookmin University, Seoul, South Korea. He is working as Professor & Director, Institute for Industry and International Programmes at Vellore Institute of Technology, Vellore, Tamil Nadu, India. He has produced two PhD students and his research area focus on Image Processing, Data Analytics & Distributed Systems.