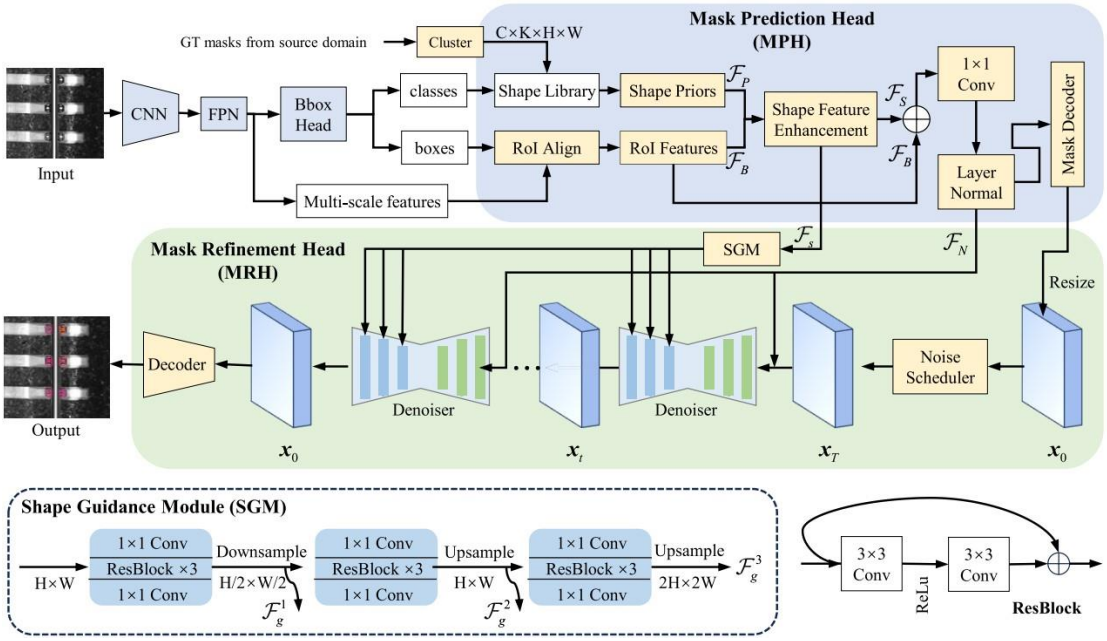
Portfolio

1.AI intern-defect detection



To address the issue of defect detection. The challenge arises due to the domain shift between training and testing environments, which can severely degrade detection performance when the model is exposed to unseen data.

Key components of the proposed framework:

Base Detector: The paper uses standard object detectors as the base (e.g., Mask R-CNN or YOLACT), which predicts bounding boxes for the fasteners.

Mask Prediction Head (MPH): This module generates the initial mask within the bounding boxes, incorporating shape priors from a shape library. The shape priors are extracted from the training data using clustering and serve as a guideline for the initial segmentation.

Shape Feature Enhancement Module (SFEM): SFEM enriches the mask prediction by aligning the feature map of the predicted bounding box with relevant shape features from the library.

Mask Refinement Head (MRH): Using a Denoising Diffusion Model (DDM), the initial mask is progressively refined to match the ground truth more closely. The refinement is carried out in an iterative manner, treating mask improvement as an optimization problem.

Shape Guidance Module (SGM): This module further enhances MRH by introducing multiscale shape guidance during the denoising process to ensure alignment with the geometric properties of the fasteners.

2. Core Innovations

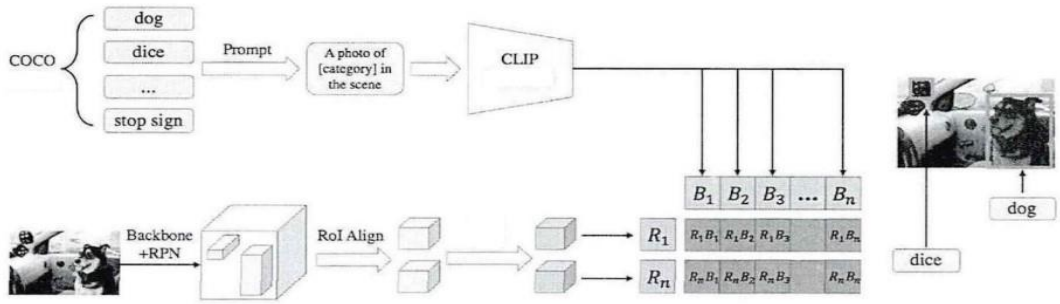
Shape-Prior Integration: The method integrates geometric shape priors, which enhances the generalization capabilities of the model, especially in unseen environments where visual features (lighting, background, etc.) may vary but the general shape of fasteners remains constant.

Denoising Diffusion Model (DDM): This novel application of a generative model (typically used

for image generation tasks) helps in refining segmentation masks iteratively, significantly improving segmentation quality, especially under domain shifts.

Plug-and-Play Nature: The MPH and MRH modules can be integrated into any standard detection framework without altering the underlying object detection structure.

2.Master Dissertation-Open vocabulary Object Detection



1. COCO Dataset and Prompts:

The algorithm leverages the COCO dataset, which is a large-scale object detection, segmentation, and captioning dataset containing categories such as "dog," "dice,".These category names (like dog, dice, etc.) are converted into prompts by adding descriptive phrases like "A photo of [category] in the scene".

2. CLIP Text Encoder:

The textual prompts generated from the COCO categories are input into CLIP's text encoder. CLIP is a model trained on large-scale image and text pairs to align visual and textual embeddings. The text encoder converts each prompt (e.g., "A photo of a dog in the scene") into a high dimensional textual embedding. These embeddings represent the semantic meaning of the category in vector space, allowing comparison with visual features later in the pipeline.

3. Object Detection Using RPN:

The input image is processed using a Backbone + RPN (Region Proposal Network). The backbone is typically a deep neural network (like ResNet) that extracts high-level features from the input image. The RPN generates several Region Proposals (R_1, R_2, ..., R_n), each representing different regions of interest in the image where objects might be located. These regions are likely to contain objects that match the categories.

4. RoI Align and Visual Feature Extraction:

The Region of Interest (RoI) Align operation ensures that the region proposals are properly aligned with the feature maps extracted by the backbone. This ensures accurate extraction of visual features for each region. These aligned features are then passed through further convolution layers to refine them, creating a set of region features (R_1, R_2, ..., R_n).

5. CLIP Visual Encoder:

While the textual prompts were previously encoded by CLIP's text encoder, the corresponding visual regions (R_1, R_2, ..., R_n) are now encoded by CLIP's visual encoder. This encoder transforms the visual data into a visual embedding that shares the same feature space as the text embeddings.

6. Matching Visual and Text Embeddings:

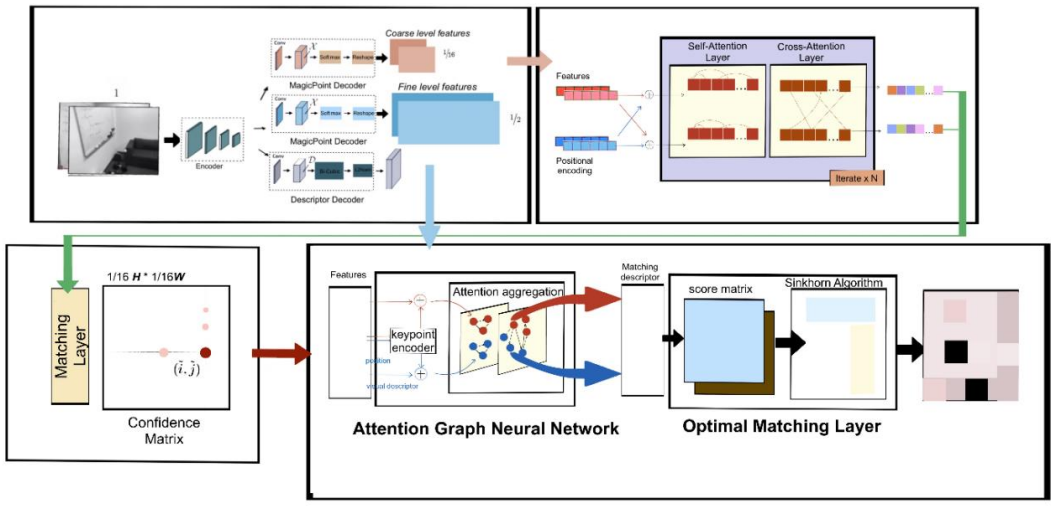
Once both the textual embeddings (B_1, B_2, ..., B_n from the prompt) and the visual embeddings (R_1, R_2, ..., R_n) are computed, they are compared to determine the best match between the detected regions and the text labels. The embeddings for the regions and the prompts are aligned. For example, a region might be identified as a "dog" or "dice" based on the highest similarity score between visual and textual embeddings.

7. Output:

Finally, the algorithm outputs the recognized objects (like “dog” and “dice”) and marks their respective locations within the image based on the region proposals. As shown in the image, the identified objects such as the "dog" and "dice" are visually tagged and localized within the scene.

3.Final year dissertation-image matching

Image matching algorithms usually consist of following steps: key point extraction, feature of key point extraction and feature matching. However, adjacent points in one image are not correctly matched to similar regions in another image of the image pair. I tried to improve the original algorithms.



This architecture can generate coarse level features and fine level features at the first step and then obtain coarse level matching based on mutual nearest neighbor. Then we can extract fine level feature points based on coarse level feature using SOTA image matching algorithms such as SuperGlue.

4.Data analysis-data churn prediction

Project Content:

Explored data analysis, missing data situations, and conducted targeted missing value imputation. For critical features with few missing values, the Random Forest imputation method was used.3-sigma rule and box plot analysis were applied to handle outliers, and categorical variables were encoded.

Employed variance analysis, removed some low-significance features through feature selection, conducted WOE (Weight of Evidence) binning, visualized the binning results for each feature, and analyzed the relationship between each feature’s binning and user behavior biases. IV (Information Value) of each feature

was calculated for further feature selection.

Trained the Random Forest model, evaluated and exported the model results. The model was used to predict user churn and make targeted adjustments based on the results. The training process fine-tuned the model by reinforcing its understanding of user behavior biases (e.g., feature importance coefficients) to analyze the key factors influencing user churn.

The link of the primary code:
https://github.com/lyz0804/data-analysis-of-the-key-factors-influencing-user-churn