Deep Learning for Industrial Applications

113034504 何懿城

1. Download the MVTec Anomaly Detection Dataset from Kaggle (here). Select one type of product from the dataset. Document the following details about your dataset:
   — I choose product transistor as the target dataset
   i. Number of defect classes: 4 defect classes and 1 good case.
   ii. Types of defect classes: bent_lead, cut_lead, damaged_case, misplaced.
   iii. Number of images used in your dataset: In below experiment, I use 60 images.
   iv. Distribution of training and test data: 48 for training and 12 for testing.
   v. Image dimensions: Each image with 3 channels. In baseline experiment, the image is resized from (1024, 1024) to (32, 32)

2. Implement 4 different attempts to improve the model's performance trained on the dataset you choose in previous question. Ensure that at least one approach involves modifying the pre-trained model from TorchVision. Summarize the outcomes of each attempt, highlighting the best performing model and the key factors contributing to its success. You may also need to describe other hyperparameters you use in your experiment, like epochs, learning rate, and optimizer. (Approximately 150 words.)

Attempt	**Epochs**	**Picture Size**	**Optimizer**	**Learning Rate**	**Pre-Trained Model**
Baseline	50	32	Adam	0.001	ResNet18
1	100	32			
2		256			
3					ResNet34
4					ResNet50

Attempt	**Train_Loss**	**Train_Acc**	**Val_Loss**	**Val_Acc**
Baseline	1.3168	47.92%	1.2443	58.33%
1	1.2428	54.17%	1.1566	66.67%
2	0.6961	85.42%	0.8620	75.00%
3	0.6695	91.67%	0.6670	83.33%
4	0.6196	93.75%	0.6933	91.67%

- Since this multi-classification is highly imbalanced, I take a percentage of 2:1:1:1:1 to get data from original dataset to prevent model only predict the same class (class with the most data)
- In above experiment, learning rate and optimizer are fixed as 0.001 and Adam
- In first attempt, I set epochs to 100. Although validation accuracy rises about 8 percent, but the training accuracy shows that the model is still underfitting.
- In second attempt, I adjust picture size from 32 to 256, and from training accuracy we know this adjustment makes the model fit the data much better, so I think this adjustment is one of the key factors.
- In third attempt, I select ResNet34 to be the pre-trained model, and both training accuracy and

validation accuracy are better than second attempt.

- In fourth attempt, I select ResNet50 to be the pre-trained model. and both training accuracy and validation accuracy are slightly better than previously attempt.
- Hence, I reckon that the key factors of best performance experiment are the size of picture and the pre-trained model selection

3. In real-world datasets, we often encounter long-tail distribution (or data imbalance). In MVTec AD dataset, you may observe that there are more images categorized under the 'Good' class compared to images for each defect class. (Approximately 150 words.)
   - Define what is long-tail distribution
     - Certain classes (head classes) have a large number of samples, while others (tail classes) have significantly fewer samples
   - Identify and summarize a paper published after 2020 that proposes a solution to data imbalance. Explain how their method could be applied to our case.
     - Paper Name: Balanced Contrastive Learning for Long-Tailed Visual Recognition
     - Methodology
       - ➢ Class Mean Contrast: Encourages embeddings to be close to their corresponding class centers, reducing intra-class variance and improving feature quality for tail classes.
       - ➢ Class-Aware Sampling: Adjusts the sampling probability to ensure tail classes are more frequently sampled during training, balancing the contrastive pairs.
     - Experiment result:
       - ➢ Achieved state-of-the-art performance on several long-tailed benchmarks such as CIFAR-10-LT, CIFAR-100-LT, ImageNet-LT, and iNaturalist2018.
       - ➢ Demonstrated significant improvements in classification accuracy for tail classes without degrading head class performance.
     - How to apply to this case:
       - ➢ Use contrastive learning with BCL to learn robust representations of both 'Good' and rare defect classes.
       - ➢ Apply class-aware sampling to upweight defect samples during training, improving defect detection performance under data imbalance.
       - ➢ Improve feature discriminability for rare defects, helping the model distinguish subtle anomalies more effectively.
     - Reference: https://arxiv.org/abs/2207.09052

4. The MVTec AD dataset's training set primarily consists of 'good' images, lacking examples of defects. Discuss strategies for developing an anomaly detection model under these conditions. (Approximately 100 words.)
   - In previous question, I already change this binary anomaly detection to a multi-class classification problem, so the strategies will be based on this assumption.
   - The strategies can be discussed by several dimensions
     - ➢ Data: Apply data augmentation techniques such as rotation, cutout, and mixup on defect images to increase diversity and sample size for minority classes, which helps reduce overfitting. Additionally, synthetic defect generation using GANs or other image synthesis

methods can further balance the dataset.

- Loss functions: we can re-weighting loss functions (focus loss or class-imbalanced loss) to let model focus on underrepresented defect classes
- Model: we can try more pre-trained model (e.g., EfficientNet, ConvNet, DesNet, VGG, …) and fine tune them on our dataset to boost our performance.

5. For the task of anomaly detection, it may be advantageous to employ more sophisticated computer vision techniques such as object detection or segmentation. This approach will aid in identifying defects within the images more accurately. Furthermore, there are numerous open-source models designed for general applications that can be utilized for this purpose, including YOLO-World (website) and SAM (website). (Approximately 150 words.)
   - To leverage these powerful models and fine-tune them using our dataset, it is necessary to prepare specific types of datasets. What kind of data should be prepared for object detection and for segmentation.
     - To fine-tune object detection models such as YOLO-World, the dataset should include annotated bounding boxes around defective areas in the images, along with corresponding class labels. These annotations are typically stored in formats like COCO JSON or YOLO TXT.
     - For segmentation models like SAM, pixel-level masks are required, which precisely delineate the shape and extent of each defect. These can be represented as binary mask images or in formats compatible with COCO segmentation standards.
   - Why are these models suitable for fine-tuning for our custom dataset?
     - These models are particularly suitable for fine-tuning on custom datasets like MVTec AD due to their strong generalization capabilities and flexibility. YOLO-World, for instance, is designed for efficient and accurate object detection, making it effective at locating various types of defects in a single image.
     - SAM offers high-quality segmentation results even when training data is limited, which is especially valuable in industrial scenarios.
     - What's more, since the MVTec AD dataset already includes high-resolution PNG mask files that accurately mark defect regions, SAM can be fine-tuned directly with minimal data preparation, enabling precise boundary detection and supporting more detailed visual inspection.