ASSIGNMENT 2B: ANOMALY DETECTION USING ANOMALIB

Zhiyuan Wang *CS-GY 6313 B*

Listing 1: Using anomalib in Python

```
from anomalib.data import MVTec
from anomalib.models import Patchcore, EfficientAd
from anomalib.engine import Engine

# training
datamodule = MVTec(train_batch_size=1,category='tile')
model = Patchcore()
engine = Engine(max_epochs=5)

engine.train(datamodule=datamodule, model=model)

# predicting
res=engine.predict(model=model,data_path ='./datasets/MVTec/tile/test/')
```

1 Usage

Anomaly detection is a vital task in numerous applications, including quality control, security, and medical diagnostics, where recognizing deviations from normal patterns can prevent significant issues. In this task we input an image to a model to let it tell if there exists anomaly patterns and where are the anomaly parts.

Anomalib provides a set of easy-to-use api for users to access the MVTec dataset as well as load pretrained weights to existing models just as the code shown above. These code by default download MVTec data and pretrained weights to local directories. Using data_path as a parameter other than already managed datasets is also OK in trainning and predicting process.

2 Model design and application

2.1 Patchcore

The PatchCore[2] model offers a highly efficient and effective approach to detecting anomalies in image data. It extract features from the input using a pretrained CNN network(or an encoder in the original paper) ,subsample the features and store them to a memory bank to reduce the complexity of the model, thus speed up the training and predicting process.

2.1.1 Feature Extraction

Feature extraction in PatchCore is central to its ability to identify anomalies. The process involves leveraging deep learning models to capture rich, high-dimensional representations of image patches, ensuring that both local and global patterns are considered. By utilizing pretrained CNNS that are adept at extracting hierarchical features, capturing complex visual patterns intrinsic to the data. These features from CNNs have wide reception fields, containing the information about a certain Patch. After dividing each input image into smaller, overlapping patches, the model generates a feature vector representing its unique characteristics. This granularity allows for detailed analysis of localized regions, increasing the likelihood of detecting subtle anomalies that might be overlooked when considering the entire image at once.

The PatchCore model effectively combines advanced feature extraction with strategic coreset selection to deliver a powerful anomaly detection solution. By leveraging the strengths of CNNs for feature representation and using a coreset to streamline processing, PatchCore achieves both efficiency and accuracy. Its ability to detect anomalies in a variety of settings makes it a valuable asset in the field of machine learning, particularly for applications requiring precise and reliable anomaly detection.

2.1.2 Coreset Selection

A coreset is a strategically selected subset of data that captures the essential characteristics of the entire dataset. In PatchCore, the coreset plays a pivotal role in making anomaly detection efficient by reducing the complexity of the feature space without sacrificing performance.After extracting features from each patch, PatchCore aggregates these into a comprehensive pool. The model then selects a representative subset from this pool to form the coreset. Methods like K-means clustering are employed to identify diverse and representative feature vectors. By selecting centroids of clusters, the model ensures that the coreset encapsulates the variation present in the dataset. After all these constructions. The selection process uses distance metrics such as Euclidean distance or cosine similarity to evaluate feature relevance. These metrics ensure that the constructed coreset maintains the diversity and discriminative power of the entire dataset.(the original paper uses euclidean distance)

2.2 EfficentAD

The EfficientAD[1] model presents a novel approach to anomaly detection by integrating efficient learning paradigms and robust detection mechanisms. This report examines how EfficientAD achieves anomaly detection, with a specific focus on its Lightweight Student–Teacher framework, Logical Anomaly Detection, and Anomaly Map Normalization.

2.2.1 Lightweight S-T framework

EfficientAD employs a lightweight student-teacher architecture, leveraging the principles of knowledge distillation to improve anomaly detection performance while maintaining efficiency. The teacher model is typically a more complex CNN neural network. It captures intricate feature representations from the input data, serving as a knowledgeable guide for the subsequent student model. The student model is designed to be lightweight and computationally efficient. Through knowledge distillation, the student learns to mimic the output behavior of the teacher model. This process involves training the student model on soft labels generated by the teacher, which contain more informative gradients than hard labels. The student model requires fewer resources, making it suitable for real-time anomaly detection in resource-constrained environments.

2.2.2 Logical Anomaly Detection

The original paper used an autoencoder for learning logical constraints of the training images and detecting violations of these constraints. The autoencoder is trained to predict the output of the teacher. In contrast to the patch-based student, the autoencoder must encode and decode the complete image through a bottleneck of 64 latent dimensions. On images with logical anomalies, the autoencoder usually fails to generate the correct latent code for reconstructing the image in the teacher's feature space. Hence the original paper doubled the number of output channels of their student network and train it to predict the output of the autoencoder in addition to the output of the teacher.The student learns the systematic reconstruction errors of the autoencoder. At the same time, it does not learn the reconstruction errors for anomalies because these are not part of the training set. This makes the difference between the autoencoder's output and the student's output well-suited for computing the anomaly map. The combined anomaly map thus contains the detection results of the student–teacher pair and the detection results of the autoencoder–student pair. By sharing the student's hidden layers in the computation of these detection, the model maintains low computational requirements, while enabling the detection of structural and logical anomalies.

The original paper also normalized the local(student-teacher) and the global(student-encoder) anomaly map to make them comparable.Otherwise, noise in one map could make accurate detections in the other map indiscernible in the combined map. The original paper achieved this by a linear transformation whose parameters are trained on validation set.

2.2.3 Application

The EfficientAD model combines a lightweight student-teacher framework, logical anomaly detection, and anomaly map normalization to create an effective and efficient anomaly detection system. Its design principles make it particularly valuable for applications requiring both precision and scalability.

3 Results

Using Api from anomalib, I test the result of Patchcore and EfficentAD on the required categories. The results (in terms of pixel level AUROC) is listed below.

Category	PatchCore	EfficentAD
tile	0.948	0.796
leather	0.990	0.938
grid	0.980	0.848

References

[1] Kilian Batzner, Lars Heckler, and Rebecca König. *EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies*. 2024. arXiv: `2303.14535 [cs.CV]`. URL: `https://arxiv.org/abs/2303.14535`.

[2] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. *Towards Total Recall in Industrial Anomaly Detection*. 2022. arXiv: `2106.08265 [cs.CV]`. URL: `https://arxiv.org/abs/2106.08265`.