Pushing baselines for outlier detection on MVTec

How far can we push classical baselines for outlier detection on the MVTec dataset?

- Outlier detection methods reach very high AUROC scores of >99% (https://paperswithcode.com/sota/anomaly-detection-on-mvtec-ad)
- How do classical machine learning methods compare to state-of-the-art methods?

Classical baselines (https://scikit-learn.org/stable/modules/outlier_detection.html):

- One class SVM
- Isolation Forest
- Kernel Density Estimation
- Local Outlier Factor
- Elliptic Envelope

Using the raw image pixels with classical baselines has two major problems:

- Curse of dimensionality: high dimensional input → very slow training
- Methods that are only able to make linear decision boundaries will most likely perform much worse than models that are able to make non-linear decision boundaries (e.g. neural network approaches). https://arxiv.org/abs/2206.06602

Therefore we use a neural network as a "feature extractor" and apply the classical approaches on the feature space.

- Pretrained networks on ImageNet (e.g. ResNet18, ResNet50, ViT-T, ViT-S, ViT-B, …) https://github.com/rwightman/pytorch-image-models
    - Pretrained networks don't have to be trained by yourself, pretrained checkpoints (on ImageNet) can be downloaded from the above git repository
- Randomly initialized neural networks

MVTec dataset is a relatively small dataset with only 100s of samples.

- Do data augmentation techniques improve the results?
- How much more runtime is required (augmentation adds additional samples → classical methods typically scale bad with lots of samples)?
- Is the runtime overhead worth the (potential) performance boost?
- Pointers for augmentation techniques used by other outlier detection techniques:
    - DifferNet https://arxiv.org/abs/2008.12577
    - FastFlow https://arxiv.org/abs/2111.07677
    - PatchCore https://arxiv.org/abs/2106.08265

Task description:

- Implement a training pipeline to train and evaluate (AUROC) classical outlier detection methods on MVTec
- Conduct experiments with different feature extractors
    - Pretrained networks
    - Randomly initialized networks
- Conduct experiments with data augmentation
- Conduct hyperparameter search for classical baselines and present results