基于正样本的缺陷检测

—— lyl

在给定只有正样本的数据情况下，训练我们的模型，输入一张带有缺陷的图片使得模型能够检测出图片中的缺陷 mask。

难点：样本少，基于正样本训练

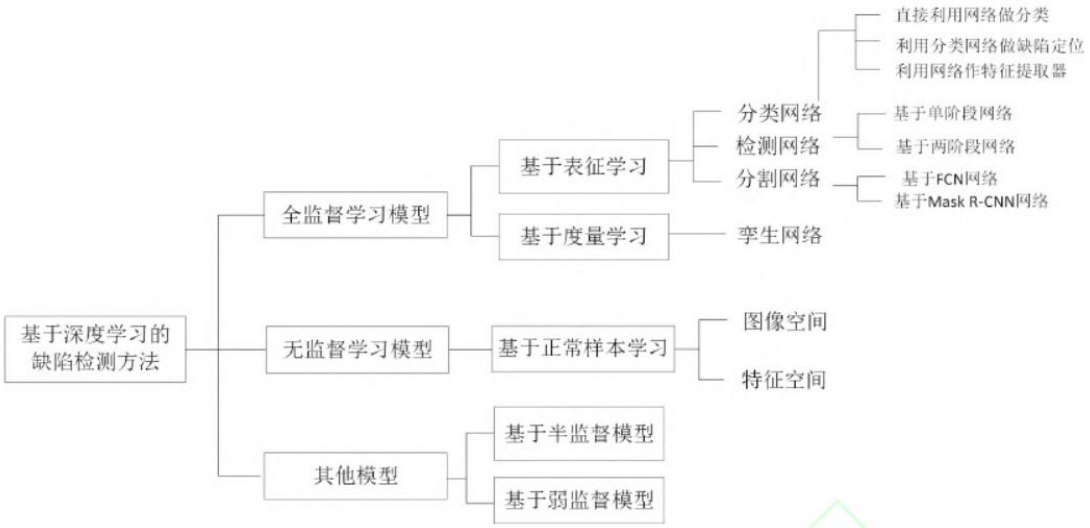


图3　缺陷检测方法框架图

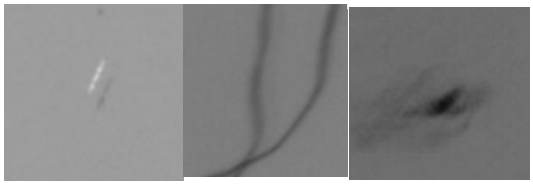
一、初赛阶段

1、初赛数据集

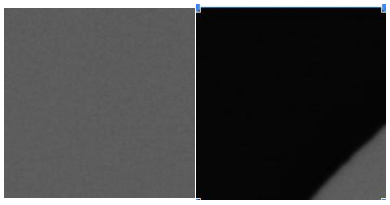
给了两个 part 的数据，part1 包含黑灰图，part2 有白边包围和纯黑图，这些都为正样本。此外赛题包含 TC_image，测试集，包含了正常样本和负样本，然后判定哪些为负样本，并将负样本中的缺陷 mask 标记出来。负样本包含了凸起，块状，线状、缺口等缺陷。

初赛背景比较单一，图片属于平坦区域，因此使用较简单的模型。

Part1 瑕疵图

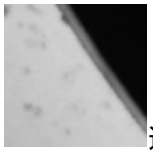


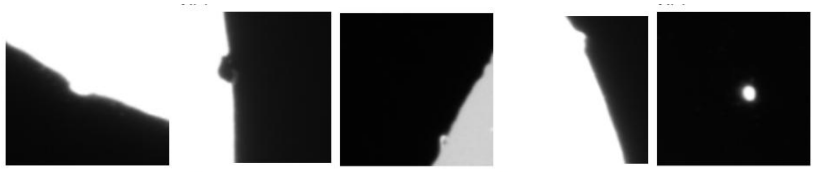
Part1 正常图



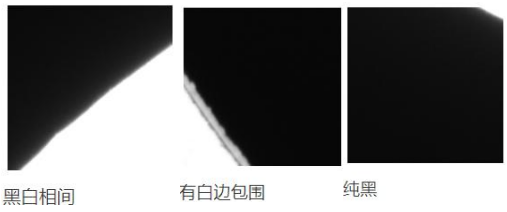
Part1 中缺陷主要是条状，块状和白斑，part1 的正常样本有两种，一种是纯灰的另一种是黑灰相间的。

Part2 缺陷图

 这种不是缺陷，由白边包裹的内部才算缺陷，外部不是缺陷。



Part2 正常图



2、 尝试的方法

首先，思考了利用 deep svdd 分辨正负样本，之后利用边缘检测检测出负样本中的边缘，之后与边缘检测中同样的做法得到缺陷坐标。

deep svdd:

方法	特征数	训练论数	n u	object	a u c
	8	200		one-class	0.89

Canny 边缘检测：利用 canny 检测边缘，设定两个不同的阈值，一个低一个高，用于检测出不同的深浅的边缘，正常边缘一般深度高，容易检测，瑕疵边缘深度低，难检出，所以阈值较高的边缘用以检测正常边缘，阈值低的正常与不正常的边缘都检出，二者相减，取出剩余的像素，将像素值相加，大于阈值设定为负例子，低于则为正例子。利用边缘检测检测出边缘，之后使用轮廓检测，得到检测出来的缺陷的轮廓，将轮廓作为缺陷坐标输出。

边缘检测高阈值	边缘检测低阈值	剩余像素阈值	Score
（180，200）	（0，50）	20	0.687739

用正例来训练变分自编码器，之后将测试数据输入到网络中重新构造，如此可以将缺陷数据的缺陷修复，之后将重新构造的数据与原有数据做差，取得差值大于阈值的像素点，若像素点的数量大于某个阈值则将该图片设定为缺陷样本。

设定为缺陷样本像素阈值	Score
part1 80 part2 1500	0.77

将自编码器与 svdd 融合

由于自编码器无法很好地重新构造出黑白相间的图片,而 svdd 能够较好地区分二者,因此将两个模型 融合,将自编码器中有二 svdd 中没有的那些去除掉,以此来模型融合

利用分类网络,将图片分为四类,纯灰,黑白,黑灰,带条纹四类,其中前三类中都可能包含缺陷,而最后一类则肯定不是缺陷,对于纯灰,自编码器有较好的检测结果,对于后两类,svdd 有较好的检测结果,因此先用分类网络对测试图片进行分类,分类之后纯灰图片送入到自编码器中处理,黑白与黑灰的图片送入到 svdd 中处理,最后带条纹的图片则 直接设定为正例子,不送到任何网络中。

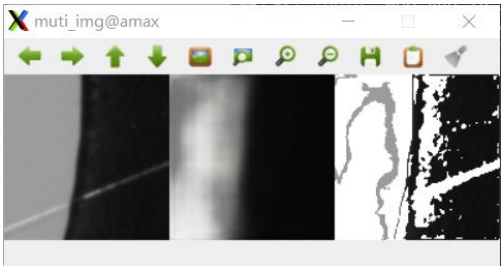
3、 我们的想法

最开始尝试用卷积网络提取 8 维特征，然后用 one-class svdd 来分类，用边缘检测提取点。初赛提交分数 0.75 分。之后尝试 deepsvdd。

用 GAN 来对原始图像随机生成一些缺陷，然后这样既有正样本又有负样本，把这样的数据放进模型训练，之后进行修复缺陷图像，在将修复后的图像与原始图像做差，得到缺陷区域。

问题：

网络难以收敛，并且制作出来的生成图与原图相差过大，难以定位缺陷。使用 GAN 修复出来的图像分辨率不高，很难去做差。



那种灰色和黑色交界的图是正样本，但是模型会把很多正样本判断成负样本。

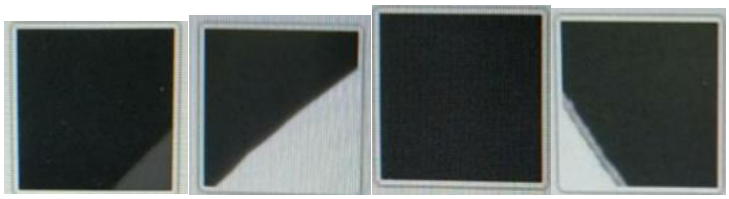
Svdd

这种方法是传统缺陷检测 svdd 的一个改进版本，svdd 为单类的 svm，只给定正样本，使得模型描述出正样本的分布，若输出样本的分布与描述的分布相差过大则视为负样本，否则视为正样本。Deep svdd 的基本思想是描述输入的样本，利用卷积神经网络使得输入的特征距离尽量小，在训练的过程中得到样本特征的平均值，通过衡量输入样本的特征与特征平均值的距离来判定样本为正样本还是负样本。

这里的代码有两种判定方法，一种是 soft-bounding，即软边界，边界距离在训练时得到，另一种是硬边界，边界由手动给定。前一种方式得到的边界效果很差，auc 接近 0.5。后一种方式则对数据很敏感，泛化能力很弱，不过对于同分布的图像能够有不错的分类效果。

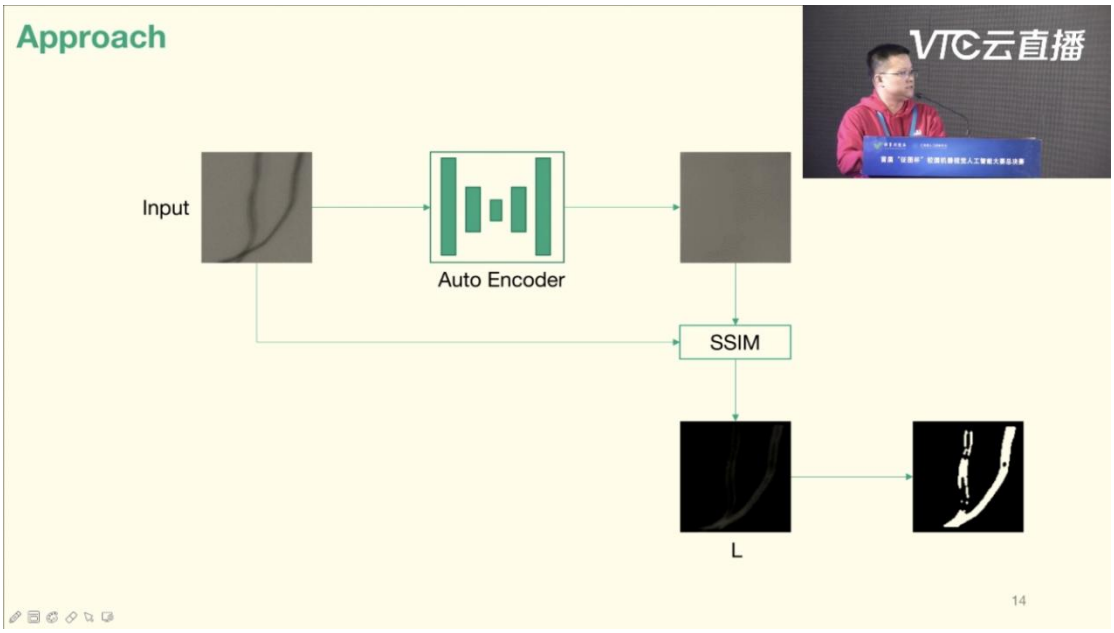
使用 svdd 得分 0.75

把图片分为纯灰的、黑白的、灰黑的以及带条纹的，



训练一个分类网络来把输入的测试图片分一下类，纯灰的交给自编码器，黑白盒黑灰的交给 svdd，svdd 判定为缺陷图的话，在把他交给自编码器来分割。如果分类为条纹的，则判定为正例。Svdd 区分黑白的还是比较有效的。

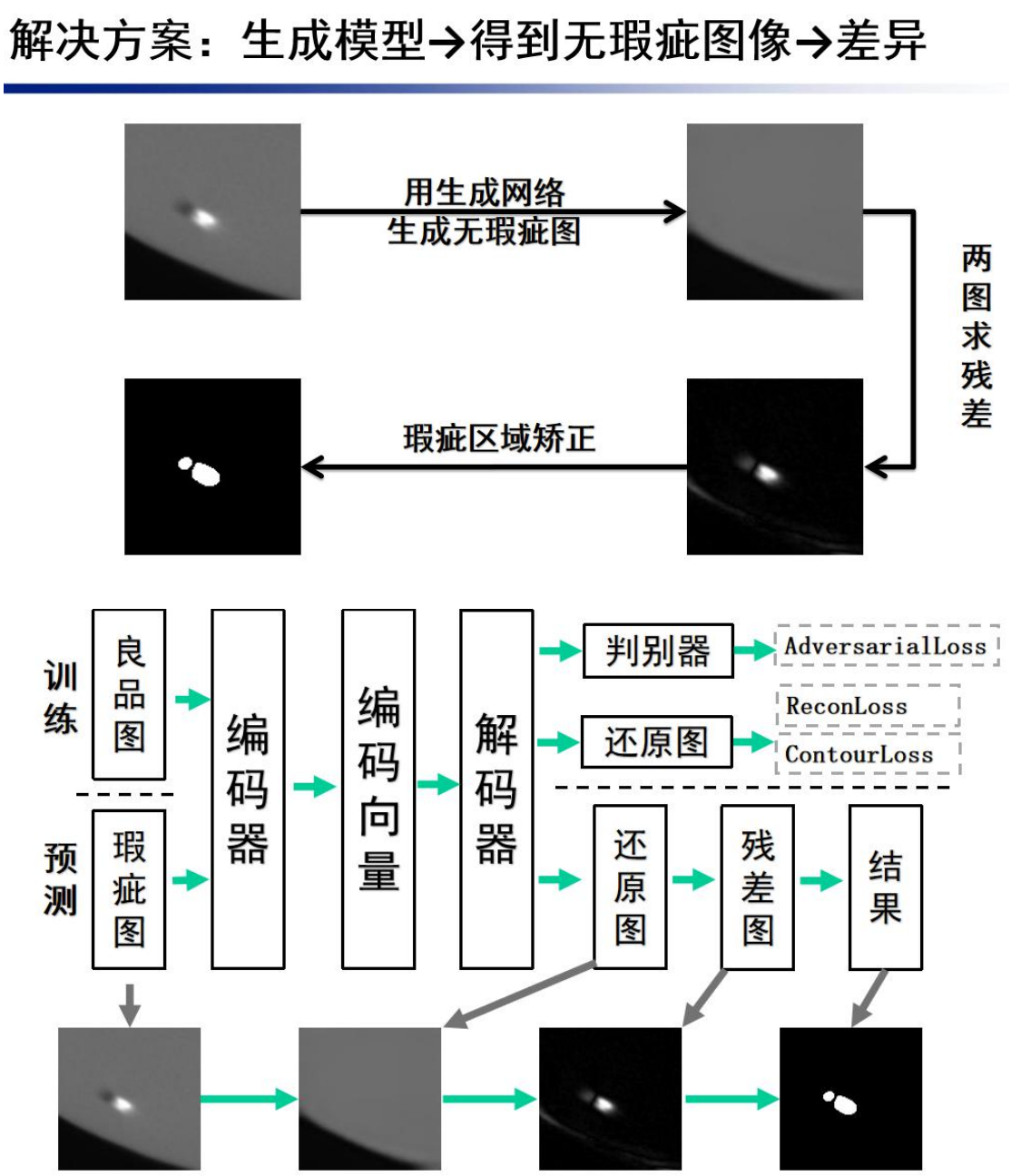
利用编码器提取图像的特征，再用解码器还原图像，由于模型只学习了正样本的特征，所以当网络输入负样本时，模型会将图片中的瑕疵去除，还原得到最终的正样本图片，将还原后的图片与还原之前的图片相减，差异较大的即为缺陷位置，根据缺陷大小来确定该样本是否为缺陷样本。



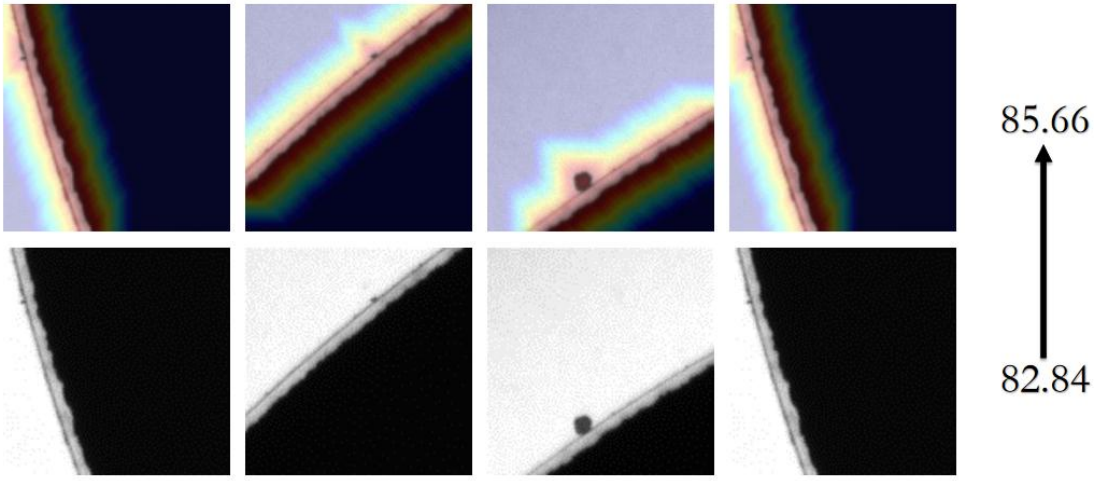
SSIM Loss

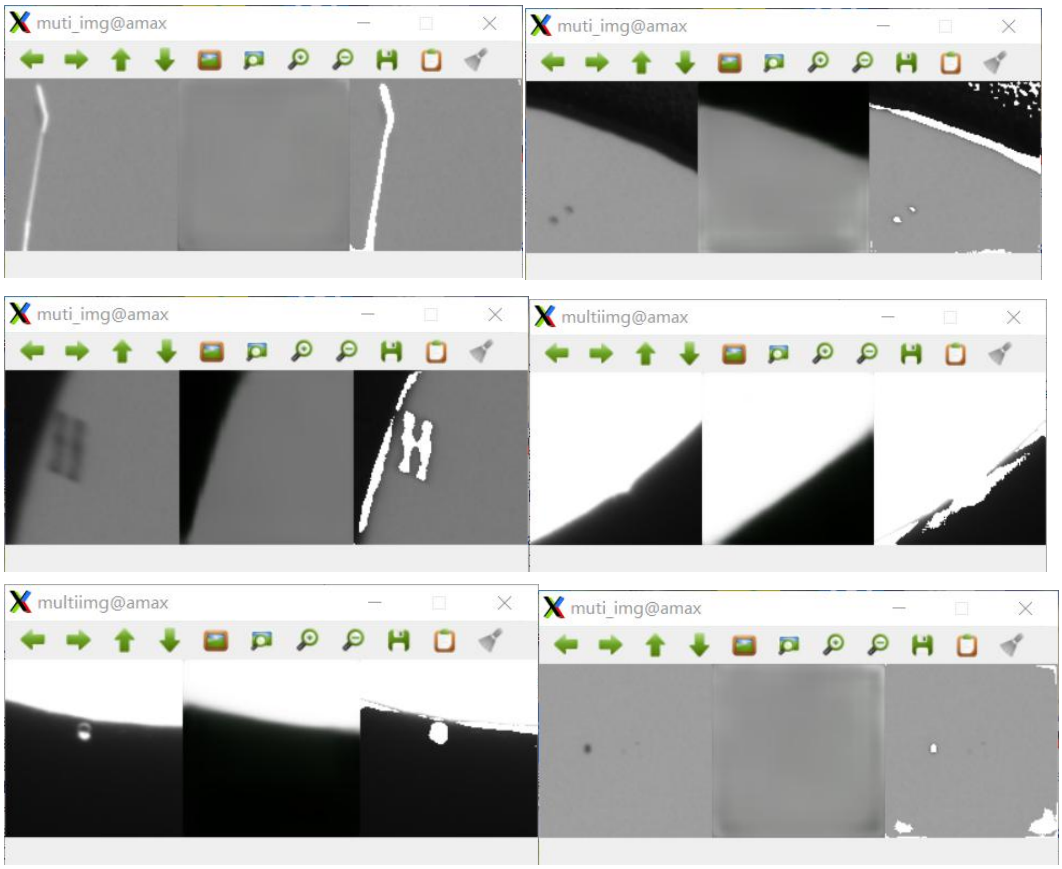
Image Quality Assessment: From Error Visibility to Structural Similarity 。该文献提出了一种取代 MSE， 衡量重建图像和原图的相似性的 metric：Structural

Similarity (SSIM)。



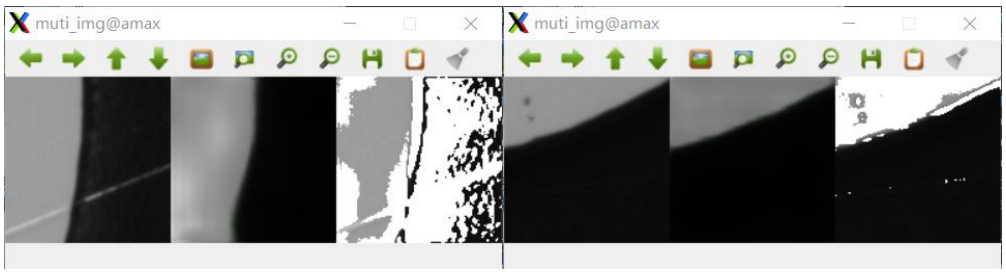
我们使用自编码器的想法与上图相同 ,在还原图该队伍加入了 ContourLoss 损失函数，利用 ContourLoss 让网络更关注结构特征，减小边缘重构误差、增强边界附近结构特征。

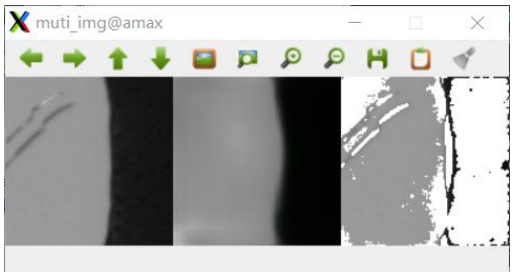




将阈值调低一点，使得产生更多的点，使得误检率低一点。

调整阈值后





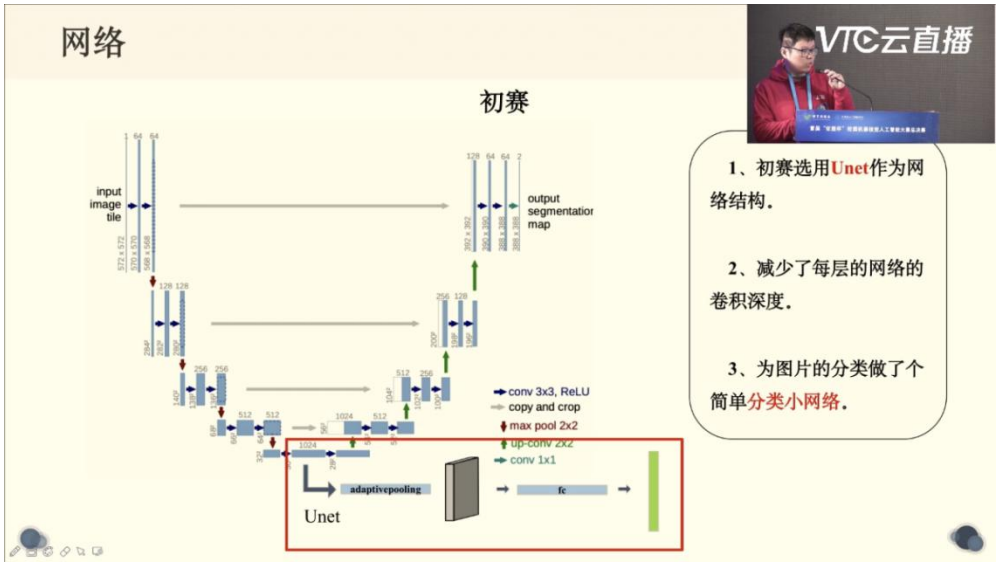
算法训练一个模型仅仅需要 5 分钟，模型简单，编码器部分包含 5 个卷积层和 5 个池化层，解码器利用上采样的方式还原图像，更大程度地保留图像原有的纹理信息，模型推理三万张图片仅需三分钟左右。

最后提交初赛出分后发现模型鲁棒性不行，很多正负例子分不开。

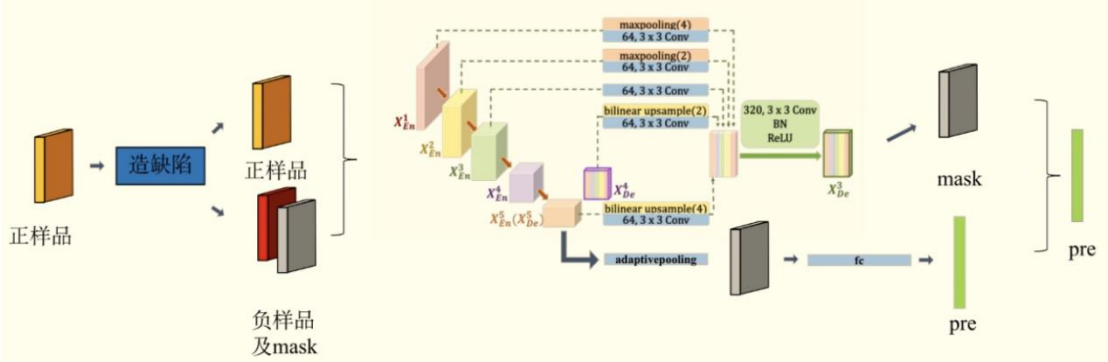
实验发现，svdd 融合到自编码器不是很有效，直接使用 wae 自编码器。

本次我们主要采用的是无监督学习的方法，而其他队伍采用的是有监督学习的方法。

4、 初赛冠军团队解法



Unet 和自编码器很像，不同的是 Unet 是全卷积网络，左侧为编码器，右侧为解码器，编码器有四个子模块，每个子模块包含两个卷积层，每个子模块之后有个通过 maxpool 实现的下采样层。解码器包含四个子模块，分辨率随着上采样操作依次上升，直到与输入图像的分辨率一致。Unet 网络还使用了跳跃连接，将上采样结果与编码器中具有相同分辨率的子模块的输出进行连接，作为解码器中下一个子模块的输入。



他们将正样本造缺陷后将正样品和负样品的 mask 输入到 Unet 网络中，经过特征提取网络，将提取的特征经过反卷积和上采样处理从而生成预测缺陷的图片，将提取的特征经过自适应池化层和全连接层进行处理得到预测缺陷的分类效果，将预测分类结果与预测的缺陷图片得到缺陷分类结果和 mask 区域。

有监督要比无监督的效果好，只要缺陷造的好。

他们提供了造缺陷的方法：

1）弹性变换 将图片进行非线性变化，将形状规则的直线，圆形等生成一些不规则复杂的曲线和块状缺陷。

2）灰度渐变 将所生成的缺陷在灰度值上有个渐变的过程，使其逼真真实缺陷，增加模型的泛化能力。

5、算法要求以及目标

仅有正样本可用

对多变瑕疵的泛化性和鲁棒性

易于训练

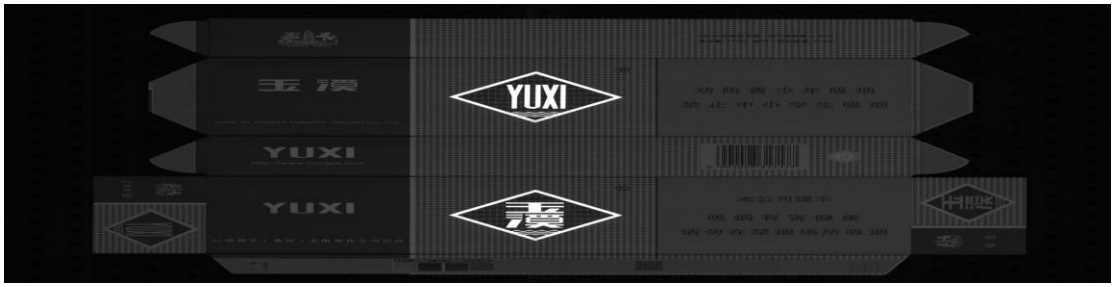
推理速度快

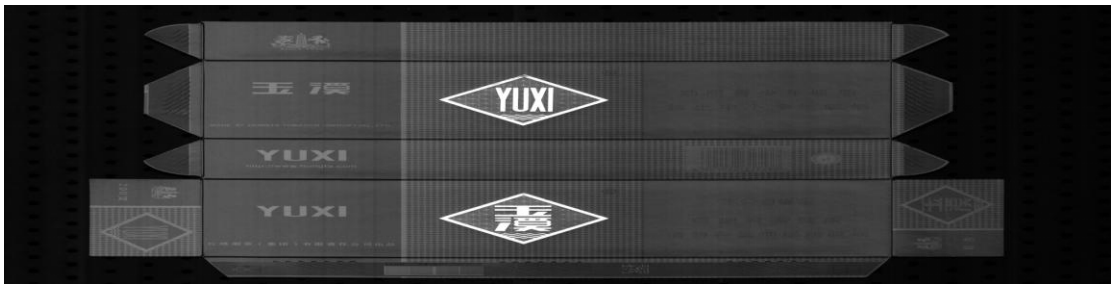
二、复赛阶段

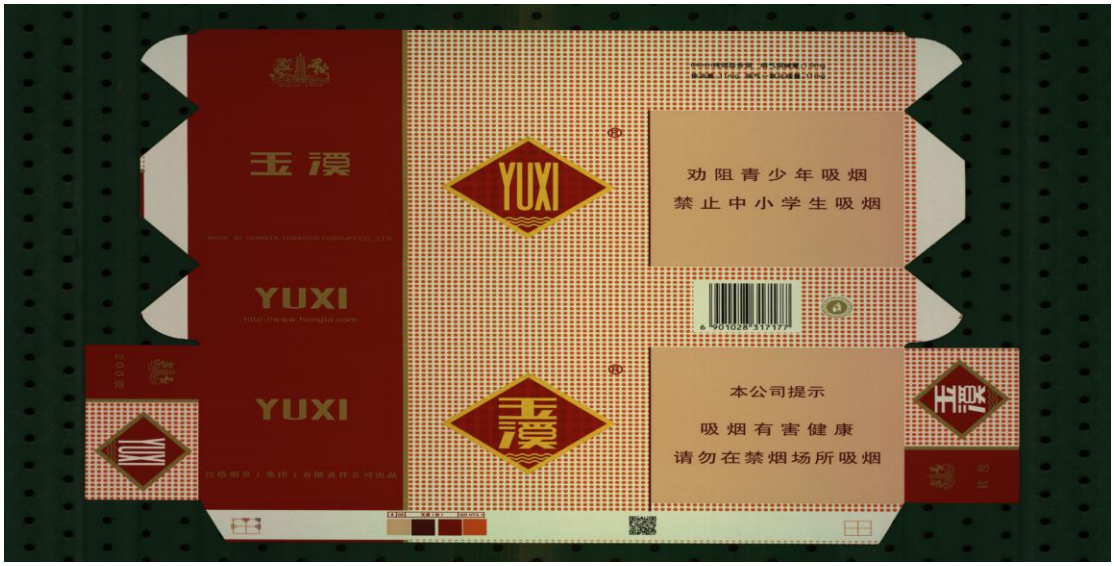
1、复赛数据集

复赛数据集训练集包含无缺陷大图原图（OK Image）而不是初赛的切片。由于复赛数据集的复杂纹理使得初赛的模型不在适合复赛的数据集。待测试的图（TC_image）是一些切片。

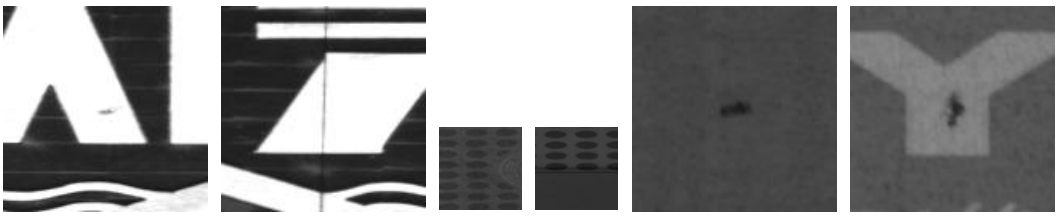
下图为 OK_images 好品原图样本：

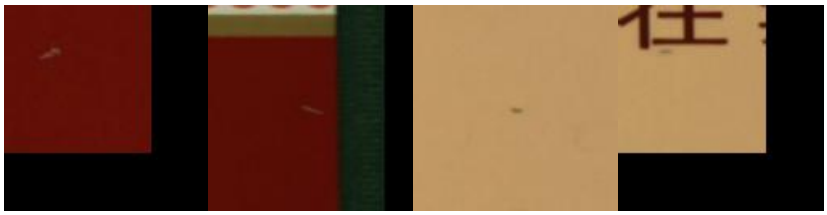






下面示例是 part1、2、3 部分的切片：





2、初赛简单的模型范式存在的问题

(1) OkImage 中有一些像缺陷的部分容易混淆

(2) TC_images 中有一些逻辑错误

(3) 网络需要强行记住每一处的复杂纹理细节。

3、方法

3.1 我们的方法

复赛基于弱监督语义分割，首先训练一个分类器，在原始好图上先随机生成缺陷，然后利用此数据训练分类网络。分类网络训练好之后，将输入图片输入网络中，首先判断是否有缺陷，如果有缺陷的话可以得到该类图片的类激活图，通过对 heatmap 设定阈值来得到最终的分割图。与此同时，还采用了多尺度推理，即将输入图像采样成不同分辨率大小，然后将得到的类激活图进行叠加用来提高准确率。除此之外，还采用了 CFR 进行修正，但是因为比赛平台不支持这个包，所以后面就没有用了，用了之后效果更好，基本完全匹配缺陷点。

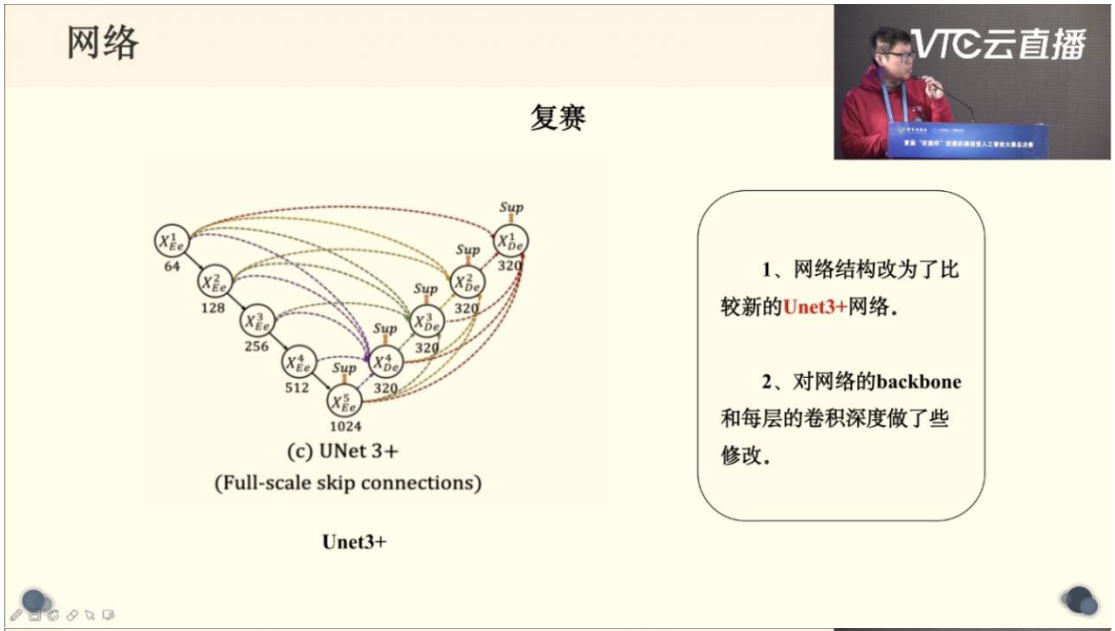
算法训练仅需要训练一个分类模型即可，推理过程快，模型计算开销小，可以做到实时检测。

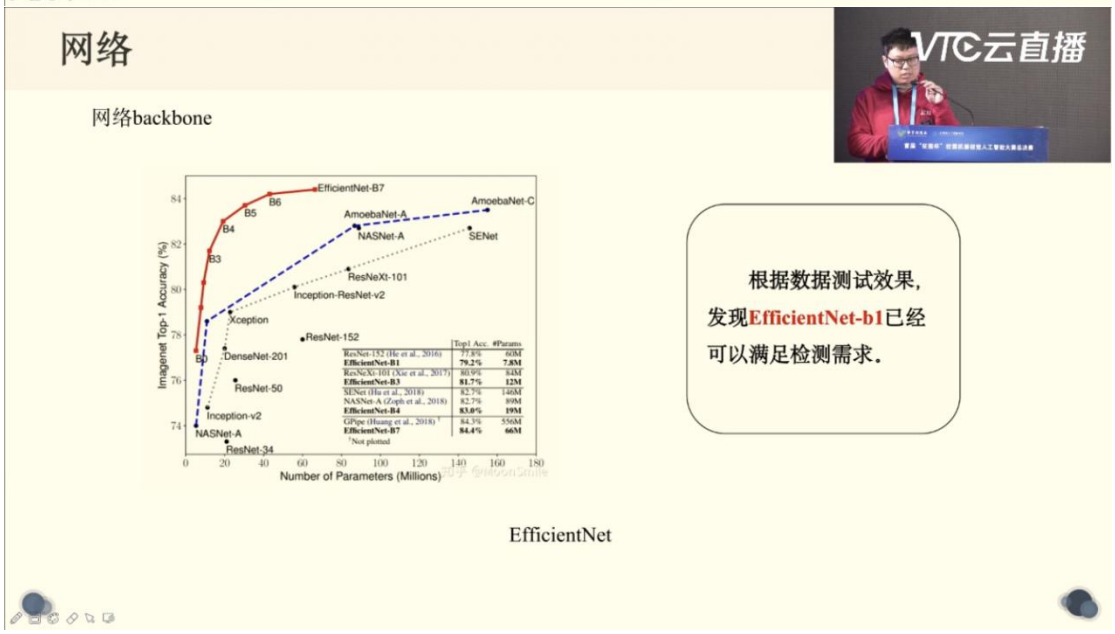
复赛借鉴论文思想：2020NeurIPS-Causal Intervention for Weakly-SupervisedSemantic Segmentation

Github：https://github.com/ZHANGDONG-NJUST/CONTA

3.2 其他方法

（一）冠军团队的解法



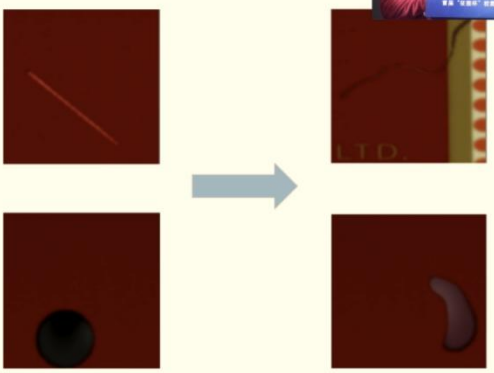


数据增广

随机翻转、色彩增广、高斯噪声、模糊、随机平移

课程学习（Curriculum Learning）

主张让模型先从容易的样本开始学习，并逐渐进阶到复杂的样本和知识.



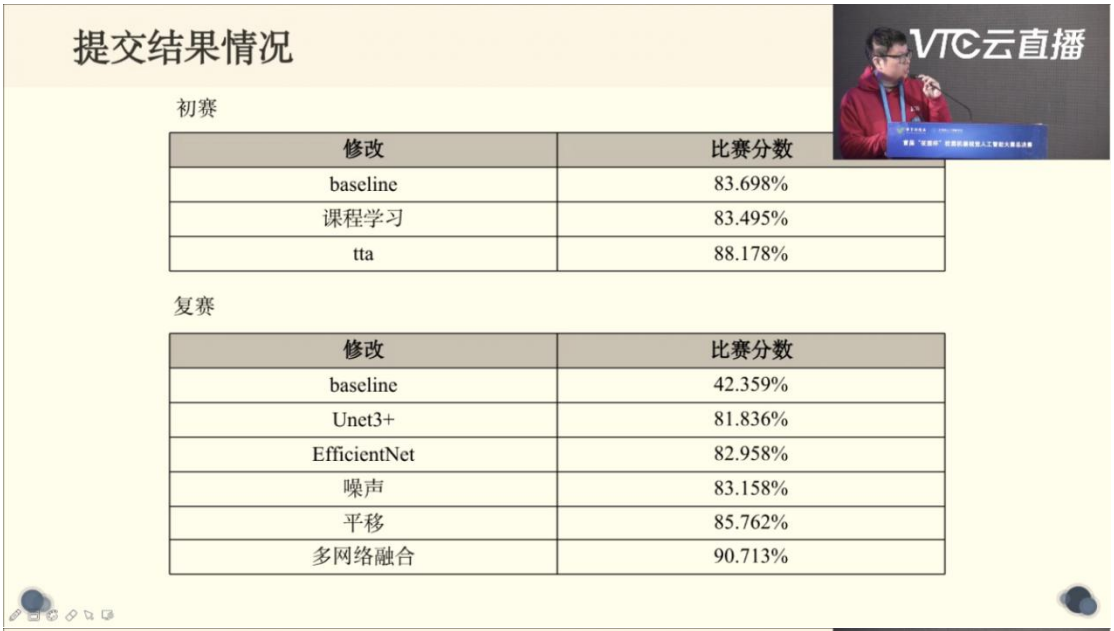
结果融合

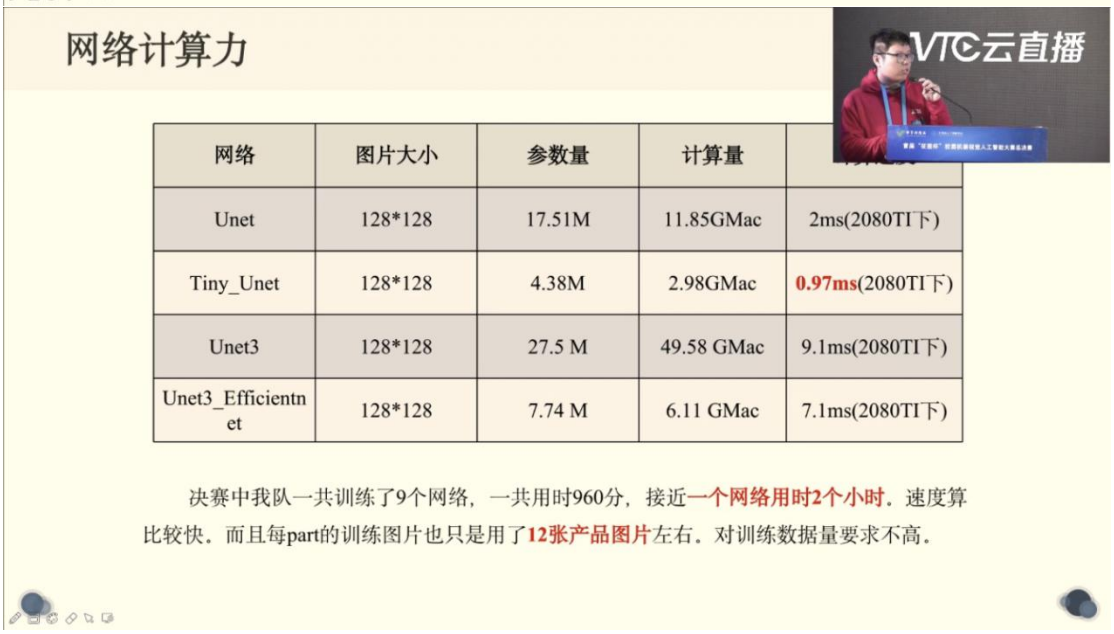
- **分类结果和分割结果融合**
- **TTA(test time augmentation)**

  用到了四个方向的翻转，色彩变化和高斯噪声或模糊等六个增广方法做处理，通过平均法或投票法获取单个网络最终结果。
- **多网络融合**

  也是采用平均法或投票法融合，提高网络鲁棒性。

- 可以选用投票法，通过调整他们融合的阈值来**调整漏检误检的效果。**



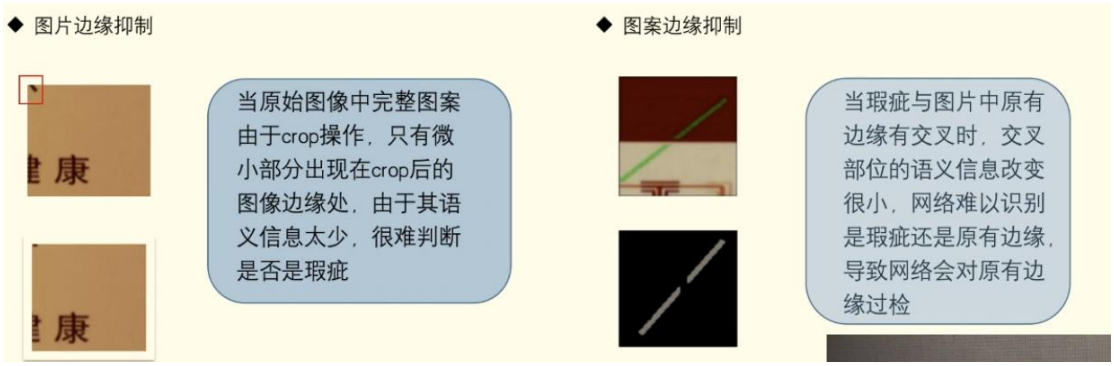


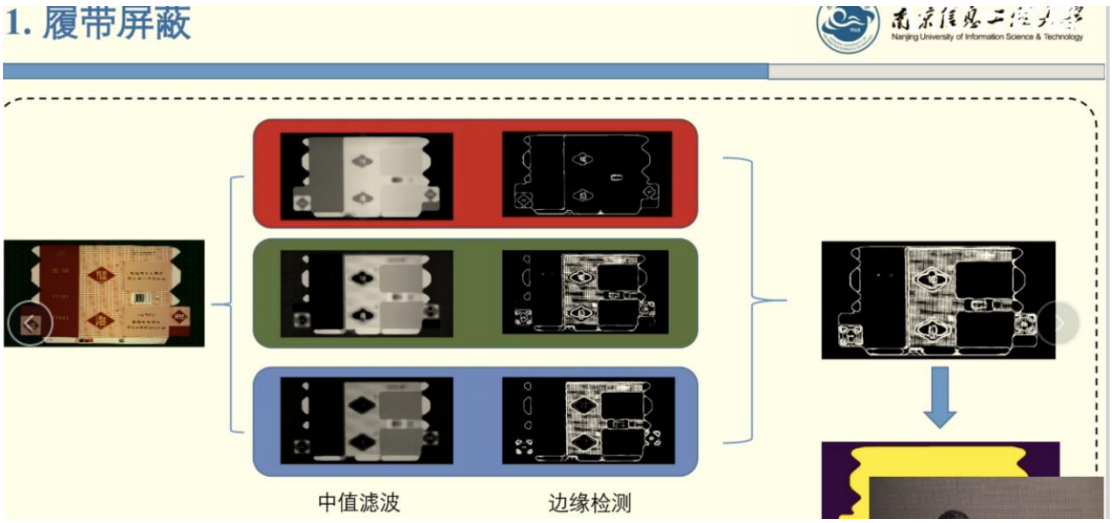
（二）对原图进行 crop 操作和瑕疵生成

对图像进行履带屏蔽，对三通道操作， 然后将大图 crop 成 128*128 小图，加入一些瑕疵，得到瑕疵样本训练模型，得到分割结果。瑕疵的位置、形状、大小、颜色均应为随机的且满足真实情况。针对瑕疵的位置，形状大小是通过二元高斯分布的均值和方差来控制。在 crop 出的小图上随机选择一个点，将该点坐标作为二元高斯分布的均值，在随机选择方差，通过公式生成了一些随机的点，这些点都是以该点坐标周围的点，对该点作为包罗和填

充，颜色是在 0-255 随机填充即可。

由于是在原图中 crop 的图，有微小部分出现在 crop 后的图像边缘处，由于语义信息太少，很难判断是否是瑕疵，因此会导致过检。





其他思路：

1）快速模板匹配卷积网络。

基于 QATM 方法得到测试图像在 OK 图像上的大致区域，在通过平方差匹配法得到准确位置。

损失函数：dice loss

Dice loss 能动态调整正负样本不均衡问题，适用于缺陷检测

Dice loss 广泛用于医学影像分割，与这里的检测任务十分相似

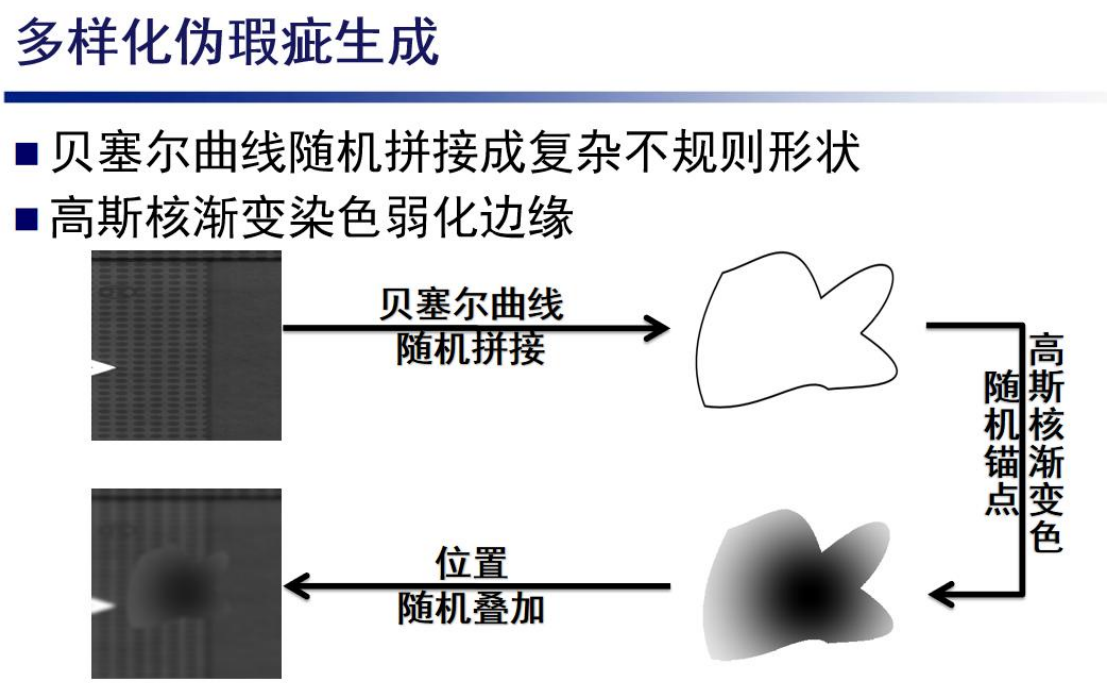
3）Student-teacher

4）孪生网络

三、总结

1、 造瑕疵方法

1）



2）手工设计缺陷、泊松融合

5）弹性变换 将图片进行非线性变化，将形状规则的直线，圆形等生成一些不规则复杂的曲线和块状缺陷。

6）灰度渐变 将所生成的缺陷在灰度值上有个渐变的过程，使其逼真真实缺陷，增加模型的泛化能力。

2、 初赛总结

方法一：

① 采用卷积自编码器

② 加入 attention 机制和 skip connection，提高模型性能

③ 加权 ssim loss 作为损失函数，提高边缘处的重建

④ 后处理抑制非 ROI 区域缺陷，减少误检

3、复赛总结

方法一：

① 基于 QTAM 方法得到测试图像在 OK 区域上的大致区域，在通过模板匹配精确定位

② 通过透视变换矫正倾斜，降低由于像素偏移造成的误检

③ 通过颜色空间转换去除传送带背景，降低非 ROI 区域带来的误检

方法二：

a. 训练和测试进行多种数据增强

b. 使用预训练的分类器作为特征提取器

c. 基于标准化网络将好品样本特征分布映射为标准正态分布

d. 训练时基于最大似然估计优化标准化网络

e. 测试时基于似然损失进行分类，基于梯度图进行分割

方法三：

train_x.py:

训练正样本数据集，通过 encoder 编码器 z=encoder(image)，然后在通过解码器重构出图像 x_recon，之后通过 MSELoss 计算重构出的图像与原图之间的损失。

```
Z=encoder(image)
X_recon=decoder(z)
Recon_loss=nn.MSELoss(x_recon,image)
```

---------mmd kernel loss　待理解

计算 z_fakea

Wae.py:

采用 wae 自编码器

Utils.py:

放一些常用函数

test_part1.py:

读取数据集，加载训练过的 encoder 和 decoder 器，

Data_rebuild1=encoder.forward(image)
Data_rebuild1=decoder.forward(data_rebuild1)

通过做差计算分割区域

residual = torch.abs(data_rebuild1.squeeze()[0, :, :] - image.squeeze()[0, :, :])
Thr=0.0700
point_set = residual.ge(thr)

Point_set.nonzero() 将非 0 的点的筛选出来并保存下来。

test_part2.py:

与 test_part1 相同流程测试 part2 的数据集

不同的是阈值设置为 0.2000

Thr=0.2000

参考 GitHub：

[1].https://github.com/schelotto/Wasserstein-AutoEncoders/blob/0588d4ea9268013bfc72038678d53494cb0ef34c/wae_mmd.py
[2].https://github.com/jiwoon-ahn/irn

参考文献：

[1].基于深度学习的表面缺陷检测方法综述_陶显

[2].Causal Intervention for Weakly-Supervised Semantic Segmentation
[3].2019cvpr_Weakly Supervised Learning of Instance Segmentation with Inter-pixel Relations
[4].2020cvpr-Self-supervised Equivariant Attention Mechanism for Weakly Supervised Semantic Segmentation
[5].Automatic Fabric Defect Detection with a Multi-Scale Convolutional Denoising Autoencoder Network Model

[6].基于卷积自编码器和图像高斯金字塔的布料缺陷无监督学习与检测方法.https://blog.csdn.net/hhy_csdn/article/details/84679448.

[7].工业图像异常检测最新研究总结（2019-2020）.