# Project.博士学位答辩论文

# Title: 运用深度神经网络实现磨损颗粒的智能在线检测

## Chapter1 研究意义/开展本项研究的重要性

### 1.1总起介绍

机械设备的状态监测是关于**收集、处理和分析**机械运转状态**信息**的技术，运用该类技术辅助机械设备维护保养的决策已经成为主流趋势。机械设备的健康状态监测系统一旦成熟，可降低设备运行的损伤、最小化维护保养费用、减少产能的损失以及避免灾难性事故的发生，从而为社会节省大量资源。目前，已经有多种状态监测技术被研究或投入使用，例如振动信号分析、锈蚀检测、利用超声波的声发射检测、油液分析以及磨损颗粒分析等。每种技术都有各自的优势和不足，其中一些技术因操作繁琐、过于消耗人力而无法满足工业应用的要求，仍然停留在实验室研究阶段。单独使用某项技术不可能解决所有基于状态监测的设备维护问题，根据特定的设备和任务来选择合适的状态监测技术是业内人士的共识。尽管如此，大量从业人员认为磨损颗粒分析是最有效的状态监测技术之一，原因是磨损颗粒是机械中摩擦副的直接产物，其数量、尺寸和形态特征与机械工作压力、润滑情况和磨损机理息息相关，与其他状态监测技术相比，具有前瞻性强、直接反映主要磨损机理等优点。半个世纪以来，关于磨损颗粒的研究从未停歇，大量学者对磨损颗粒的形状、材料、尺寸以及表面纹理进行了深度的探索，磨损颗粒分析技术得到了长足的发展。在检测和诊断劣化的机械元件时，磨损颗粒的特征可以提供信息并辅助用户进行决策。然而，磨损颗粒分析技术目前缺乏自动化，现阶段的应用十分依赖专家经验，且耗时长，费用昂贵，上述缺点制约了该技术在工业上大面积推广应用。尽管有这些不足之处，磨损颗粒分析依然备受关注，作为一种评估和预测机械状态的方法，磨损颗粒分析前景广阔，且一旦实现智能化、在线化，解决其耗时耗力的问题，该技术必能带来巨大的经济效益。

### 1.2研究背景

现阶段磨损颗粒分析仍大多采用离线分析的方式，存在诸多弊端：

* 为进行不同实验，样品需要在不同的容器中转移，加大了样品污染的风险；
* 检测流程繁复枯燥，消耗大量时间和人力资源；
* 若想得到设备精准的诊断结果，还需要相关专家的评估，而在实际应用中，不同专家经常得到不同的结论，缺乏一致性；
* 时效性差，样品经历采样、储存、运输、送实验室分析需要若干天，实验结果生成时，被检测设备可能已经失效。
* 浪费时间和金钱，实际应用中，绝大部分离线分析都是多余的，因为结果都是：没有异常。

离线式磨损颗粒分析的局限性推动了该技术自动化、在线化的发展。自动化的磨损颗粒在线分析可以规避许多上述缺点，首先，在线采样不会让样品暴漏在外界的污染源下，降低了样品被污染的风险；其次，自动化分析过程减少了人类对结果的干预，增强了结果的一致性，也节省了宝贵的人力资源；最后，在线分析不存在时效性问题，只有当出现问题时才会触发下一步行动，不需要定期停机采样，节省不必要的开销。基于上述原因，业内人士认为磨损颗粒分析目前发展的大方向是自动化、在线化。

目前在线磨损颗粒检测仍处于研究阶段。硬件的使用上，研究中多使用小于20倍放大倍数的显微传感器，与离线分析时使用200倍显微镜获得的清晰图片仍有较大差距；算法上，研究中多采用颜色和尺寸等特征配合传统分类器对磨损颗粒大致分类，准确度仍然不高，且未出现有效的算法能够完美解决图像中自动分割问题，因而对图像中的颗粒进行自动计数、准确分类依然没有实现。

人工神经网络（Artificial Neural Network）是对生物神经网络的一种模拟和近似，是由大量神经元相互连接而构成和自适应非线性动态网络系统。上世纪50年代末，Rosenblatt提出了单层感知器模型，第一次把神经网络的研究付诸实践，但该模型不能解决线性不可分问题。直至1986年，Rumelhart和Hinton等提出了一种反向传播算法（Back Propagation），才解决了感知器的问题。由于当增加神经网络的层数时，传统神经网络会遇到局部最优、过拟合以及梯度扩散等问题，神经网络的研究一度止步不前。2006年，Hinton等人在《Science》上发文，其主要观点有：1）多隐藏层的人工神经网络具有优异的特征学习能力；2）可通过“逐层预训练”来有效克服深度神经网络在训练上的困难，从此引出了深度学习（Deep learning）的研究，也掀起了人工神经网络的又一热潮。目前，常用的深度学习模型有对抗性生成神经网络、自动编码解码机、卷积神经网络（CNN）等，已经广泛应用于图像、语音识别、翻译、自动驾驶、智能机器人等领域。

深度神经网络在图像识别任务中主要运用的模型是卷积神经网络（CNN），卷积神经网络的局部连接、权值共享及池化操作等特性使之可以有效的降低网络的复杂度，减少训练参数的数目，使模型对平移、扭曲、缩放具有一定程度的不变性，并具有强鲁棒性。早在上世纪80、90年代，CNN就在手写数字识别中表现优异。然而此时的CNN只适合做小图片的识别。直至2012年，Krizhevsky等使用拓展了深度的CNN在ImageNet大规模视觉识别挑战竞赛（ImageNet Large Scale Visual Recognition Challenge，LSVRC）中摘取桂冠。自此，在大小图像识别赛事中，夺冠的均为深度神经网络模型。而基于CNN，深度神经网络在图像分割任务中的效果近年来也极大的改善。这些进步主要归功于Mask/Faster RCNN以及Fully Convolutional Network（FCN）等基本框架的提出和发展。目前相关研究的主要致力于解决即时分割问题，即时分割的挑战性在于准确的检测到图像中的所有目标且能准确地分割出每一个对象，这就结合了传统的目标检测和目标分割两个任务，前者是检测目标的类别并用围盒（bounding box）进行定位，后者则侧重于将图像中的每一个像素进行分类标记。

综上所述，深度学习在解决图像分割和图像识别等任务的能力十分强大，且已经在许多领域进入实用化阶段，应用于磨损颗粒智能分析具有广阔的前景。然而，到目前为止，国内外关于研究深度神经网络实现在线磨损颗粒智能检测的研究基本为空白。因此，应开展深度神经网络实现磨粒图像智能检测的研究，开发适用于磨粒图像智能分割和自动分类的深度网络总体结构、与之配套的训练算法以及模块结构、权重初始化方法、迁移学习方法等细节问题，解决磨损颗粒图像背景复杂、颗粒重叠等制约磨损颗粒智能识别的瓶颈问题，最终形成一整套适用于磨损颗粒的智能分析流程。为以后的机械摩擦状态在线监测装备设计提供方向性和系统性理论依据。

### 1.3研究目标

1.提出在线磨损颗粒检测的创新点。

2.通过一系列试验建立适用于磨损颗粒图像快速且高精度检测的深度神经网络结构，为磨损颗粒自动分割和识别提炼出有效模型。

3.优化网络的训练方法，研究迁移学习对分割和识别的影响，完成准确分割并识别疲劳块状磨粒、层状颗粒、严重滑动磨粒以及滑动滚动混合颗粒的任务，为在线磨损颗粒检测提供技术支持。

### 1.4论文结构

本篇论文的结构安排如下：第二章文献综述主要介绍磨损的机理和特征、磨损颗粒的产生和分类标准、磨损颗粒智能分析的软件和硬件发展、深度神经网络的发展和原理。第三章

第四章实验

第五章

## Chapter2 文献综述/国内外研究现状动态分析

### 2.1 磨损颗粒分析

本章将详细介绍磨损颗粒的产生和分类，磨损颗粒显微图像的特征，并结合国内外研究现状和动态发展描述磨损颗粒分析面临的关键问题和相关研究结果。本章还将介绍深度神经网络的基本原理和概念，对比传统机器学习算法的优势和劣势，以及深度神经网络应用于磨损颗粒分析的前景。

### 2.2 磨损颗粒的产生

摩擦力是阻止接触物体之间相对运动的一种力，这种力在生活中随处可见：人们在路上行走是依靠鞋子和路面之间的摩擦力；冬天摩擦双手取暖是通过克服摩擦力做功产生热量；没有摩擦力我们甚至无法用筷子夹起食物[[1]](#footnote-1)。摩擦力产生的主要原因是接触表面的不规则性，在特定尺度下观察，任何表面都布满不规则的锯齿，或称‘微凸体’。经过加工的金属看上去或摸上去都非常光滑，只有在显微镜下才能观察到表面的微凸体，当两个表面接触时，实际发生接触的区域是微凸体之间的接触，一般来说实际接触面积远小于名义接触面积。两个表面的实际接触面积取决于微凸体的分布以及外部施加的载荷，载荷的增加会导致接触面积的增加，进而产生微凸体塑性变形甚至造成材料转移和脱落，也就是磨损，脱落的部分即是磨损颗粒，如图1所示。在相对运动物体表面加入润滑剂就是为了保持微凸体不直接接触从而降低摩擦。材料之间发生接触和相对运动的情况出现在许多应用场景中，例如滚动轴承、气缸和活塞环、汽车的刹车片还有金属加工和锻造等，这些材料之间的接触和相互运动都是人为设计的。也有一些情况不是人为设计的，比如在周期震荡载荷下某些结构的连接部位发生周期性位移，也被称作微动磨损（fretting），抑或是润滑油中混入的较硬的污染物和工作表面发生接触导致切削磨损等。

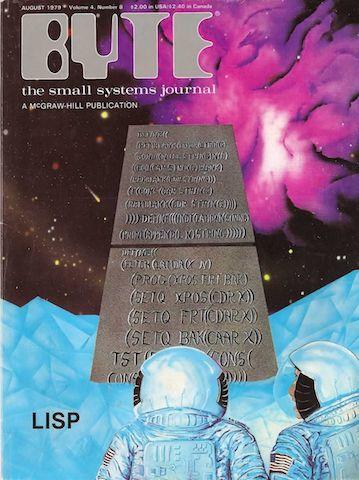


Figure 光滑的表面在显微镜下的纹理

一般机械的磨损过程包含三个时期：磨合期、正常工作期和失效期，图2-1中展示了几种机械运转周期中磨损量(磨损导致的质量损失)的变化趋势。

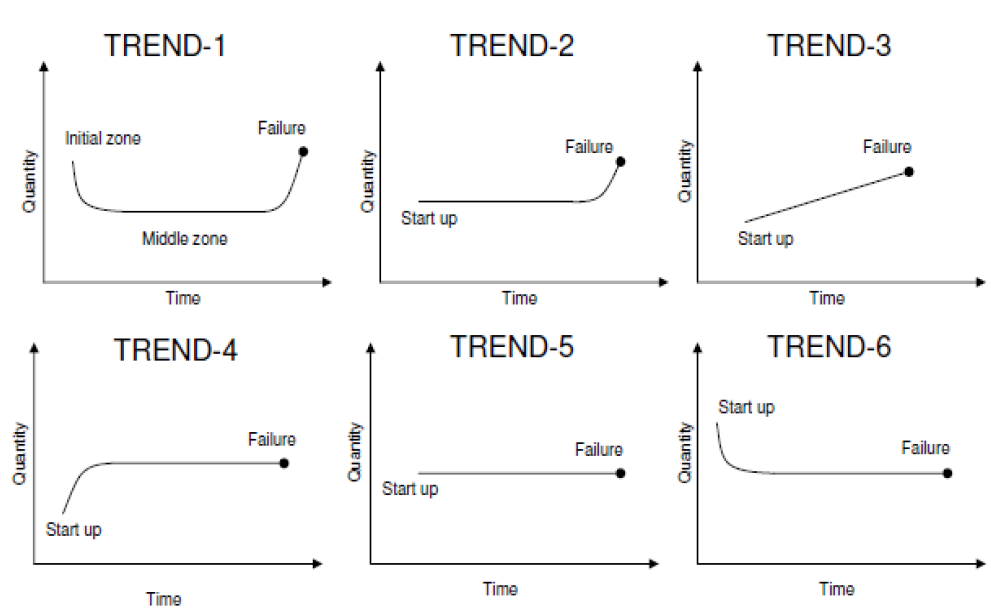


Figure 几种磨损趋势

趋势1形状如浴缸，开始的磨合期的磨损量较大，摩擦副表面的微凸体大面积脱落，随着时间推移，磨损量逐渐减小，机械进入长时间的正常运转期，此时磨损量保持稳定，最后，机械进入失效期，磨损量加大，脱落的磨损颗粒的尺寸变大。浴缸形曲线是一种理想趋势，在实际的齿轮机、内燃机和轴承中，磨损量的变化趋势很少符合此形状，在Moubray的文献报告中[],仅有4%的失效符合这一趋势，因此仅仅使用磨损量本身很难为机械的运转状态提供太多有效信息。比如图2-1中的趋势2，磨损量在很长时间内保持恒定，无法判断机械在磨合期还是正常运转期，然而一旦进入失效期，机械的状态恶化，磨损颗粒的数量和尺寸都会增加。Moubray认为2%的失效符合这一趋势。趋势3中的磨损量随着时间增长，趋势4在磨合期磨损量增长，其他情况稳定，趋势5磨损量一直维持稳定，根据Moubray的报告，这三类情况分别占5%、7%、14%，最常见的情况是趋势6，磨损量在磨合期减少然后维持恒定直到失效，这种情况占比68%。图2-2中给出了美国国家航空航天局测试的齿轮磨损的真实数据[]，显示了在250小时内磨损量变化趋势。在该图中，磨损率开始达到约0.5mg/min，然后以非常低的磨损率稳定运行，200小时之后磨损率急剧增加，表明齿轮已到达失效区域。

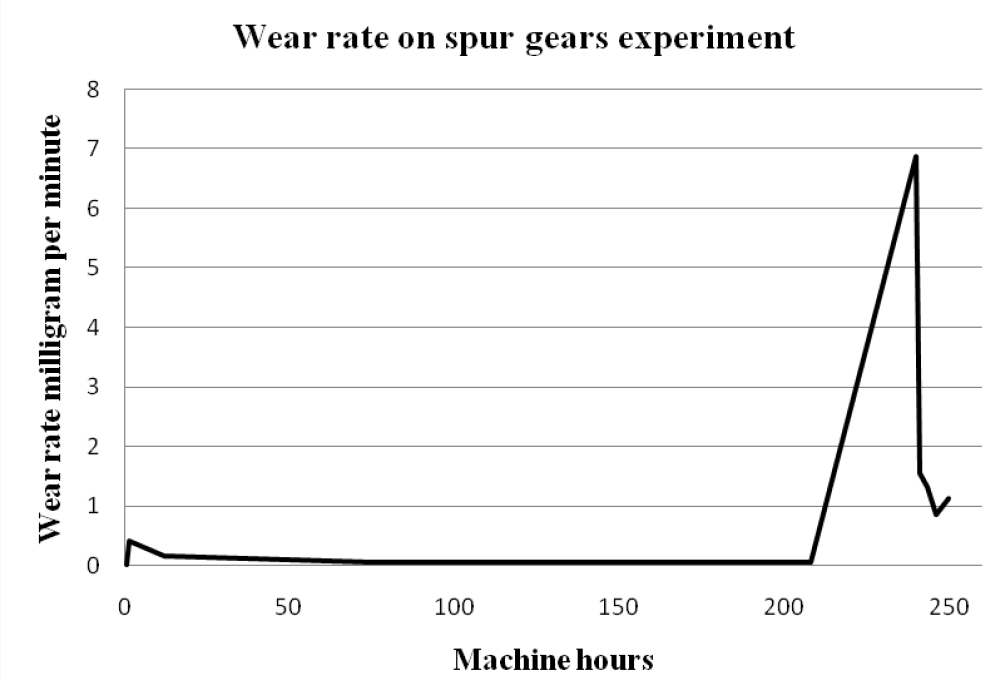


Figure 2-2

图2-4给出了机器在不同时期磨损量和磨损颗粒尺寸分布，从图中可以看出，正常磨损时，大部分颗粒都不超过10μm，接近失效区域时，机器会产生更多的大尺寸颗粒，并且保持这一趋势直到失效区域。图中还显示了光谱、铁谱和铁磁塞技术对不同尺寸的颗粒检测的敏感度。其中光谱对小颗粒检测效率高，铁磁塞主要针对超大型颗粒，铁谱则介于二者中间。本次研究中，使用改进版的铁谱，因为铁谱的检测范围为5至200μm，这个尺寸的颗粒对机器的运转状态具有重要的预测功能，可以实现早期的故障预报。光谱只能检测小颗粒，无法检测不正常磨损产生的大颗粒，而铁磁塞检测的颗粒太大，生成200μm左右的颗粒时，机械早已发生故障。

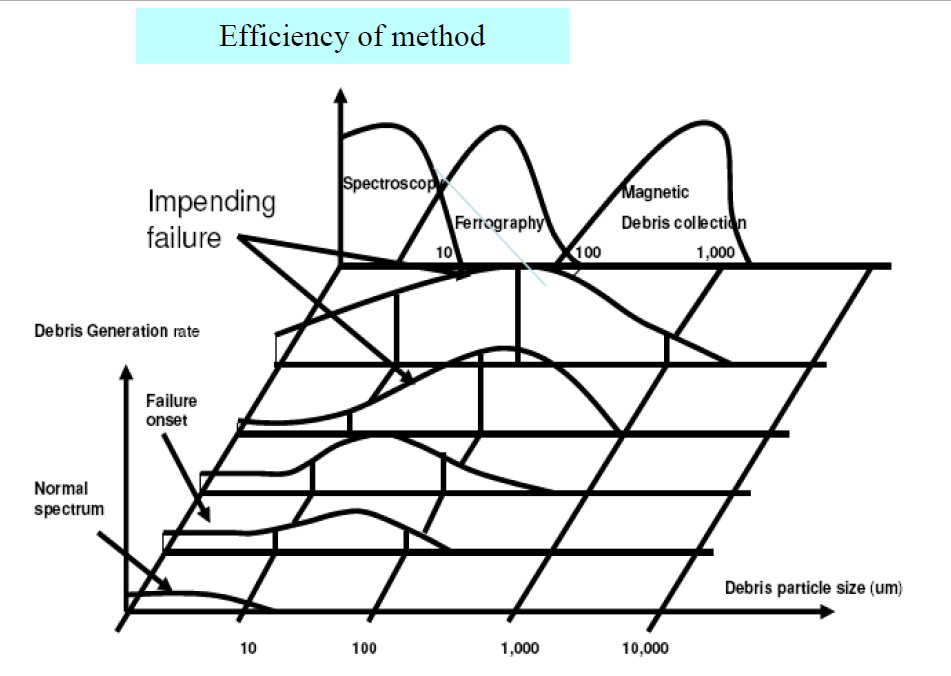


Figure 2-3

### 2.3 各类摩擦副 发生的摩擦相关现象在物理层面上的解释

磨损是两个相互接触的固体表面在相对运动中导致的表面损伤或脱落，源于表面发生的物理作用或化学作用。如前所述，大多数情况下是表面微凸体相互作用引起。描述磨损相关的现象时，很容易混淆的概念是导致磨损的相对运动和导致材料脱落的物理机理。磨损试验可以产生特定的相对运动，如滑动或滚动，但具体的磨损机理随材料、载荷和运动速度而变化。表1中列举了常见的能导致磨损的相对运动，每种情况还可以细分，例如相对滑动可以是匀速的或变速的，相对滚动可以发生在两个滚动体之间也可以是滚动体和槽道。工业中常见的摩擦部件中，滑动轴承、活塞环属于相对滑动，各类滚动轴承为滚动伴随少量滑动，齿轮和凸轮机构为滑动和滚动混合。



In order to clearly describe the generation of wear debris from the sliding and rolling action when gears mesh, information on how the meshing mechanism works is essential. A load is applied to the region of the contact point of the two meshing gears and is caused by the sliding and rolling action between them. The contact point becomes flattened elastically when the forces are transmitted. The greatest pressure is at the centre of the flattened area. Fig 2-5 shows the contact points when gears mesh.

为了清楚地描述齿轮啮合时滑动和滚动动作产生的磨屑，关于啮合机构如何工作的信息是必不可少的。 负载施加到两个啮合齿轮的接触点的区域，并且由它们之间的滑动和滚动动作引起。 当力传递时，接触点弹性变平。 最大的压力是在扁平区域的中心。 图2-5显示了齿轮啮合时的接触点。

Adhesive

If two atomically clean metallic surfaces are loaded against one another, then at points at which opposing asperities make contact, strong adhesive junctions will be formed. If the surfaces are now forced to move tangentially, not all these junctions may shear along the original material interface, but some may be deformed in such a way that fragments of the softer surface are plucked out and removed: these particles constitute adhesive wear debris. Of course, the majority of engineering surfaces are not atomically clean, nor do they always carry normal loads sufficiently high to generate enough surface plastic deformation that will lead to the formation of such adhesive junctions; consequently, significant adhesive wear is unlikely in equipment operating within normal design constraints.

如果两个原子清洁的金属表面相互加载，则在相对的粗糙点接触的点处，将形成强粘合剂接合点。 如果现在迫使表面切向移动，则并非所有这些连接都可以沿着原始材料界面剪切，但是一些可能以这样的方式变形，使得较软表面的碎片被拔出并移除：这些颗粒构成粘合剂磨损碎屑。 当然，大多数工程表面不是原子级清洁的，它们也不总是承载足够高的正常载荷以产生足够的表面塑性变形，这将导致形成这种粘合连接点; 因此，在正常设计约束下操作的设备中不太可能发生显着的粘着磨损

Subsurface cracks may be nucleated at microstructural defects or inclusions in the material, giving rise to characteristic pitting fatigue generating wear or debris particles which are more or less equiaxed.Where the friction or traction forces are sufficient to deform the surface layer, material may be lost, in the form of thin flakes or platelets. In such delamination wear subsurface plastic shear is associated with the formation and propagation of cracks nucleating from pre-existing voids or inclusions present in the material structure. The thickness of the wear sheet is controlled by the location of these subsurface cracks, and is typically of the order of a few microns. In relatively defectfree materials, for example vacuum remelted bearing steels, local incremental plastic strains can again build up, cycle by cycle, producing a form of local surface collapse, sometimes known as ratchetting, occurring on an even finer scale and producing debris which is again characteristically lamellar or sheet-like in form, but is now less than a micron in thickness.

次表面裂纹可能在材料中的微观结构缺陷或夹杂物中成核，产生特征性点蚀疲劳，产生磨损或碎片颗粒或多或少等轴。当摩擦力或牵引力足以使表面层变形时，材料可能会丢失，以薄片或薄片的形式。在这种分层磨损中，次表面塑性剪切与从材料结构中存在的预先存在的空隙或夹杂物成核的裂缝的形成和传播相关联。磨损片的厚度由这些表面下裂缝的位置控制，并且通常为几微米的量级。在相对无缺陷的材料中，例如真空重熔轴承钢，局部增量塑性应变可以再次逐周累积，产生一种局部表面坍塌形式，有时称为棘轮形，发生在更精细的尺度上并再次产生碎屑形状为片状或片状，但现在厚度小于1微米。

目前，有关磨损的分类方式没有统一的认识，主要表现在磨损模式与磨损机理没有区分，毕竟，摩擦的微观机理至今没有明确定论，也没有任何数学模型能很好的模拟复杂的摩擦过程。宏观上，磨损机理主要可分为粘着磨损、磨粒磨损、疲劳磨损，此外还有化学因素导致的各类腐蚀等[28]。

从磨粒与故障模式建立关系的角度看，从损伤机理角度分析显然更具有优势，也就是将磨损机理分为粘着、磨削、疲劳、摩擦化学等，在多数情况下，这些机理同时存在，但所占比例不同。ISO 15243.2004中把磨损机理分为了6大类，疲劳、摩擦、腐蚀、电化学腐蚀、塑性变形和断裂，细分共有15种，下面以一个滚动轴承来简单介绍一下常见几类磨损的机理。

2.2.1 磨料磨损

磨料磨损在润滑机械中是最常见的模式。它的原理是微观的金属之间发生切削，形式类似于钩犁。污染颗粒可以造成切割和损伤，凸起的表面也会对有相对运动的配对表面造成同样的切割。三体磨削是当一个尺寸和润滑油层的厚度接近的相对较硬的污染物，被相对运动的两个表面挤压时发生。当颗粒的尺寸比润滑层的厚度大时，刮擦、犁、刨削等情况就会出现。这种磨损的改善需要增强过滤和冲洗，把小颗粒隔离出去[29]。

二体磨削是因为相对运动的表面其中一个有凸起，这个凸起会直接对另一个表面进行切削。因为不是由于污染物颗粒导致的切削，所以称之为二体磨损。由此类磨损产生的颗粒有时呈螺线状，有时呈现卷状，这是因为细软的颗粒被反复碾压导致的，磨料磨损的原理示意见图2-1。这种接触主要是由于其他磨损导致的不完全润滑或者表面粗糙度太大，边界润滑时是这种接触的高峰期。要减少这种磨损，可以使用粘度高的润滑油，或者硬度高的金属，甚至安装的时候加热后消磁都是可以的。

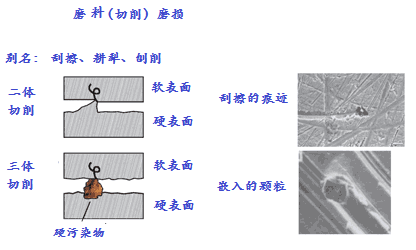


图2‑1 磨料磨损原理示意图

Fig 2-1 The schematic of abrasive wear

2.2.2 粘着磨损

粘着磨损是从一个接触表面金属转移到另一个金属表面造成的，当高压高温或者高载荷导致两个粗糙金属表面微凸体接触，由于相对运动，在表面间的接触压力作用下，发生塑性变形的同时，产生大量的热，导致这些接触点发生点焊，也被称作“固相熔焊”之后迅速分开，在小范围内发生剪切，这就是粘着磨损[2]。

点焊之后表面会变得粗糙，有时候有缺口参差不齐，有时由于形变变得相对平滑。粘着磨损主要发生在混合润滑或边界润滑的情况下，这些都是润滑供应不足，不合适的粘度，不匹配的内部间隙，不正确的安装和校准等导致的，粘着磨损在活塞环和气缸、轴承和齿轮箱中都会出现，粘着磨损的形成原理见图2-2所示。

通常的粘着磨损是温和的，轻微粘着磨损粘着点的结合强度不高，剪切应力发生在接触表面，因而不会带来太多的材料转移，而转移的金属会以微小的磨屑脱落，这就形成了上述的正常磨损颗粒。而一般粘着磨损就属于中度磨损，当摩擦副中软的金属的剪切强度小于粘着点的强度时，在距结合处不远的软金属表面内部发生破坏，软金属粘附在硬金属表面上。最严重的磨损则是金属拉毛，金属涂抹和咬死等情况，这些情况的剪切都发生在金属的深层部位，由此产生的颗粒表面也通常具有划痕。降低载荷，减少冲击载荷，保证润滑油的粘度指数则是预防严重粘着磨损的方法，解决方法之一是可以使用添加剂。

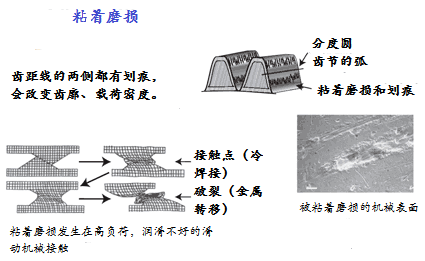


图2‑2 粘着磨损原理示意图

Fig 2-2 The schematic of adhesive wear

2.2.3 疲劳磨损

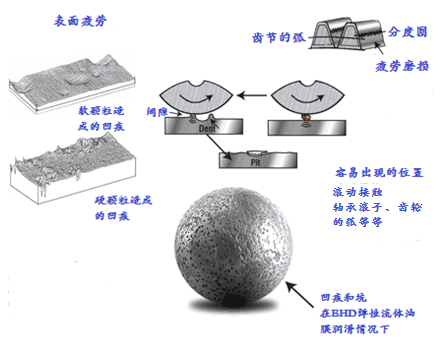


图2‑3 表面引起的疲劳磨损原理示意图

Fig 2-3 The schematic of surface-initiated fatigue wear

疲劳是几种磨损机理里面较为复杂的一种，分为表面引起的疲劳和表面下引起的疲劳两大类。

表面引起的疲劳源于润滑状态失效，普通润滑膜的破损。当滑油薄膜减少到边界润滑或者混合润滑状态，这时候金属—金属接触就会发生，相对滑动也就出现了，进而引发表面损伤。金属表面较为凸起的部分就会脱落，形成一个无光的粗糙表面。这不同于粘着摩擦造成的涂抹效应。这种表面损伤通常放大3~5倍就能观察到。

这种表面损伤是伴随着滚子对滑道的周期载荷出现的。这就造成了表面粗糙的微裂纹和微小的剥落。这些裂缝从表面逐渐扩张到金属的内部。金属表面产生的的边缘，在裂缝的边缘上弯曲。这些裂缝会扩张，也许会贯穿金属，导致一整片材料的脱落。表面疲劳也有可能是塑性变形引起的，污染物颗粒进入滚子和滑道内部或者齿轮的两齿之间，如图2-3所示。

表面下的疲劳也是由于高应力循环载荷导致金属的弯曲。这样会首先导致表面下的金属产生裂纹，然后再扩散到表面上，最终导致一片表面金属脱落。

这种情况的发生起始于金属表面以下的杂质或者失效。表面下裂纹由于长期的循环载荷应力（500000 psi）导致弹性形变而出现。这在所有的滚珠轴承和齿轮中都很常见，它们都是在弹性流体动力（EHD）工况下运作的器械。这种接触应力会集中在金属表面下的一个点上。

这些裂纹通常会扩散到表面，最终导致一整片的金属脱落。形态上和磨损产生的比较大的坑洞这种被称为剥落的情况很像。实际上，当金属的质量高一点的时候，这种情况并不常见，通常失效都是由于其他原因，良好的润滑也能预防这种情况。表面下引起的疲劳失效通常是工件已经超过了使用年限，年限是由荷载，运转速度和润滑的厚度决定的。

2.2.4 其他形式的磨损

腐蚀磨损是典型的磨损模式。根据腐蚀的介质不同可分为：氧化磨损、特殊介质腐蚀磨损、气蚀。氧气、酸碱等介质会和材料发生化学或者电化学相互作用，从而改变材料性质形成磨损，而气蚀则是液体中的气泡在高压区溃灭时产生高温形成的。

电流同样会造成磨损，局部电压过高可能会产生电弧，导致金属表面瞬间融化，形成坑洞。即使较低的电流持续作用，也可能会引起不严重的磨损。

微动磨损的机理是：摩擦表面间的法向压力使表面上的微凸体粘着。粘合点被小振幅振动剪断成为磨屑，磨屑接着被氧化。被氧化的磨屑在磨损过程中起着磨粒的作用，使摩擦表面形成麻点或虫纹形伤疤。这些麻点或伤疤是应力集中的根源，因而也是零件受动载失效的根源。根据被氧化磨屑的颜色，往往可以断定是否发生微动磨损。如被氧化的铁屑呈红色，被氧化的铝屑呈黑色，则振动时就会引起磨损。有氧化腐蚀现象的微动磨损也称微动磨蚀。在交变应力下的微动磨损称为微动疲劳磨损。

事实上，在同一种磨损过程中，往往同时存在多种磨损机理。例如，在微动磨损或磨粒磨损过程中，粘着、磨削、疲劳、摩擦化学机理同时存在，这就增加了磨损研究的复杂性。

尽管在磨损形式问题上众说纷纭，仍然有必要把它们归纳为几个基本类型，合理的分类能够使计算机识别工作简化，有利于计算机识别过程的进行。

摩擦学过程异常复杂，在摩擦学系统中，因为表面摩擦副之间的相对运动，加之界面介质和环境气氛的相互作用，最终产生形态各异的磨损颗粒。作为摩擦学系统的输出变量，磨损颗粒不仅携带了摩擦过程中的大量信息，而且其形态、颜色、尺寸等因数与磨损方式息息相关。因而磨损颗粒分析成为了定性分析摩擦副系统的主要内容。结合上一节对损伤机理的介绍以及前人的研究，本文大致将需要分类的磨损颗粒分为如下几类：

2.2.1 正常滑动颗粒

正常滑动磨损颗粒：正常滑动颗粒是机械经过磨合期之后进入稳定期，这段时间产生的最大量的磨损颗粒。在零件的正常磨损期，其表面形成一个均匀的薄层。这一薄层是约30μm的微晶结构层，称为切混层。切混层显示出极高的延展性，它可沿表面滑动数百倍于其厚度的距离。它的这种承受应力时的滑动能力将形成几乎光滑的磨痕。只要切混层是稳定的，表面即处于正常磨损。这时零件生成的磨粒称为正常滑动磨粒。它是由运动零件表面的切混层发生局部剥落而形成的。形状上，此类颗粒拥有片状的不规则轮廓，呈现薄片状，典型的正常滑动颗粒如图2-4所示。

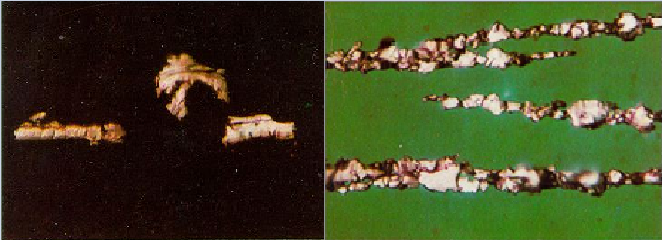


图2‑4 典型正常滑动颗粒的铁谱形貌

Fig 2-4 A typical normal sliding particle morphology

2.2.2 切削颗粒

由上文可知，切削颗粒来源于磨料磨损，可以分为二体磨损和三体磨损，无论是外部的坚硬颗粒在摩擦副软表面造成钩犁作用，还是较硬摩擦副表面有坚硬而尖锐的凸起，导致对软表面进行切削，产生的颗粒形状相似，通常会呈现为细丝、螺旋圈或者月牙形。切削颗粒特点明显，如果大量产生则意味着零件安装不良或者摩擦副中混入了较硬污染物，两种情况会严重影响摩擦系统运行，典型的切削颗粒如图2-5所示。

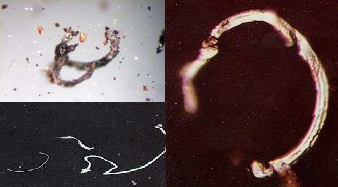


图2-5 典型切削颗粒的铁谱形貌

Fig 2-5 A typical abrasive particle morphology

2.2.3 严重滑动颗粒

严重滑动磨损颗粒：严重滑动磨损颗粒从机理上来说也是粘着磨损，但是由于润滑不足抑或是超高应力，使得稳定的混切层被破坏，形成了此类颗粒。形状上，此类颗粒拉长变大，纹理上有着明显的直线划痕，颜色则取决于磨损的严重程度和温度，如图2-6所示。

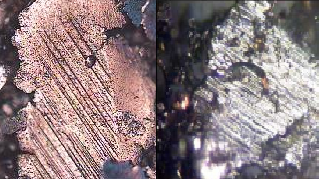


图2‑6两个严重滑动摩擦颗粒的铁谱图片，该类磨损颗粒具有明显的表面划痕

Fig 2-6 Two typical severe sliding particle with scratches

2.2.4 疲劳颗粒

微型剥落磨粒是滚动接触疲劳中早期的剥落颗粒。这些颗粒会造成许多看不见的小坑，导致表面粗糙。这些从表面脱落的材料会在经过滚动接触区域的时候被过度碾压成小的片状的颗粒。这种情况下生成的疲劳颗粒相对来说较小，长轴在10μm到30μm之间；但是，有时候长轴长度也会达到50μm。在过度碾压后，这些小颗粒的表面会比较光滑，厚度大概在长度的十分之一，甚至更薄，典型的疲劳初期的颗粒如图2-7所示。

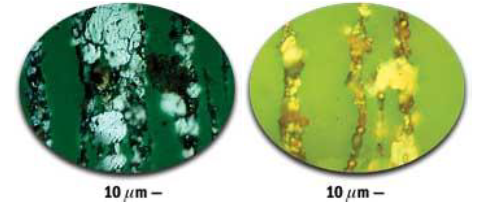


图2-7 典型疲劳初期颗粒的形貌

Fig 2-7 The morphology of typical early fatigue particles

微型疲劳颗粒和正常滑动磨粒的样子非常相似。不过对于疲劳颗粒来说，大颗粒（比10μm大的）和小颗粒（比10μm小的）的比例要比普通摩擦大得多。而且，油样里的微型剥落颗粒的集中程度比正常滑动颗粒要小。

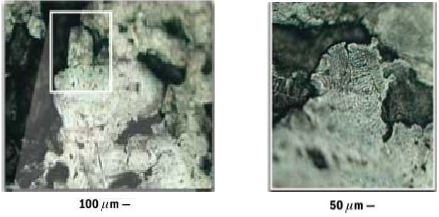
随着机械运转，金属疲劳进入中期，这时会产生层状颗粒，层状颗粒是滚动轴承里面最有特点的颗粒。层状颗粒是微型剥落情况恶化成可见的表面坑洞和剥落。在脱落之前或者之后，这些层状的材料被反复碾压，拥有以下几个独有的特色：

（1）层状颗粒是自由游离的金属颗粒，主尺度在50μm到几百微米之间，表面平整，轮廓不规则；

（2）层状颗粒时常具有孔洞和褶皱；

（3）层状颗粒具有很高的形状因子，20：1到50：1之间。

典型的层状颗粒的形貌如图2-8所示。



（a） 还未剥落的层状颗粒 （b）层状颗粒具有孔洞和皱褶

图2‑8 典型的层状颗粒形貌

Fig 2-8 The morphology of typical laminar particles

严重的深层疲劳颗粒是坑洞和剥落再进一步恶化产生的颗粒。在这个阶段，表面疲劳的裂缝已经深入扩张到表面以下，与滚动方向成大约45°角。疲劳裂纹改变方向，意味着轴承断裂的危险性很高。一般轴承到了深度剥落这一步就被认为失效了。因而，这种宽大型的深层疲劳颗粒应该作为轴承即将失效的重要预警，典型的后期疲劳颗粒如图2-9所示。

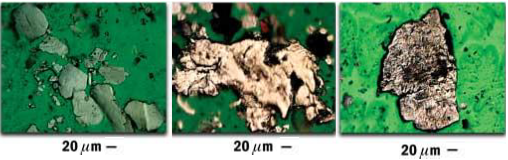


图2‑9 深度疲劳产生的颗粒形状和尺寸没有规律

Fig 2-9 Deep spalling particles have no regular shape

2.2.5 氧化物颗粒

机械在运转过程中不可避免的受到各种外界因素的影响，如空气中的水、油液、腐蚀性气体等。铁的氧化物分为红色氧化物和黑色氧化物两种。黑色氧化物是润滑不良、存在过热的标志，颗粒外观为表面粗糙不平的堆积物，边缘能透过少许光。红色氧化物是润滑系统中存在水分的标志。红色氧化物是顺磁性的，不以显著的磁性特征在铁谱片上沉积，大的红色氧化物颗粒在铁谱片上很多。

上文大致介绍了各种磨损颗粒，本文的目的则是利用不同种类颗粒的不同特征来实现智能化分类。表2-1给出了各种摩擦颗粒分类以及每种颗粒的大致特征的表格[19]。

表2-1 各种磨损颗粒的分类和初步特征

Table 2-1 The classification of wear particles and their characters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 磨粒种类 | 磨粒尺寸μm | 磨粒边界形状 | 表面形貌 | 表面颜色 |
| 正常滑动磨粒 | 不到15 | 薄片、轮廓光滑 | 大体光滑 | 金属原色 |
| 球状磨粒 | 不到10 | 球状、轮廓光滑 | 光滑反光 | 金属色占优 |
| 切削磨粒 | 二体30~100 | 细长 卷曲 | 大体光滑 | 金属色占优 |
|  | 三体<10 | 细长 卷曲 |  |  |
| 疲劳剥落 | 15~100 | 不规则边界 | 表面有小颗粒 | 同金属原色 |
| 层状磨粒 | 20~50 | 不规则边界 | 时常有皱褶 | 同金属原色 |
| 严重滑动 | >20 | 有长直边 | 直线划痕 | 金属色占优 |
| 红色氧化物 | >200 | 片状边界粗糙 | 表面粗糙 | 桔红色 |
| 黑色氧化物 | >200 | 粒状 | 表面粗糙 | 黑色 |
| 铜合金磨粒 | >200 | 片状边界粗糙 | 表面光滑 | 桔红色 |
| 铝合金磨粒 | >200 | 片状边界粗糙 | 表面光滑 | 亮白 |

### 2.4 磨损颗粒的分类

Wear debris carries with it information about its origin and the health conditions of the machine component. This information is required to be systematically and effectively analyzed to yield an appropriate action to be taken on the machinery. One of the ways to make the information gathered become meaningful is by classifying the particles into different types that can be used as a reference. Wear debris can be classified into morphological and compositional attributes. Sections 2.5.1 and 2.5.2 will briefly explain both attributes.

**2.5.1 Particle morphology**

Recent developments in WDA suggest that wear particles can be classified into at least six morphological attributes, such as a particle size, shape, edge details, colour, thickness ratio and surface texture [30]. Theoretically, the size would indicate the severity, the wear rate and the wear mode. The other morphologies could be usedto identify the source of the wear, the wear mode and the severity of the wear. The amount and concentration of wear would also assist in determining the wear rate and wear severity [30; 31]. Figure 2-7 simplifies how morphology can be used to diagnose the wear of machine components. 33

磨损碎屑带有关于其起源和机器部件的健康状况的信息。需要对这些信息进行系统和有效的分析，以便对机器采取适当的行动。使收集的信息变得有意义的方法之一是将粒子分类为可用作参考的不同类型。磨损碎片可分为形态和组成属性。第2.5.1和2.5.2节将简要解释这两个属性。  
2.5.1颗粒形态  
WDA最近的发展表明，磨损颗粒可以分为至少六种形态属性，例如颗粒大小，形状，边缘细节，颜色，厚度比和表面纹理[30]。从理论上讲，尺寸表示严重程度，磨损率和磨损模式。其他形态可用于识别磨损的来源，磨损模式和磨损的严重程度。磨损的量和浓度也有助于确定磨损率和磨损严重程度[30; 31。图2-7简化了形态学如何用于诊断机器部件的磨损。 33

**Figure 2-7: Relationship between wear characteristics and wear particles [30]**

Here, the morphology of the wear debris particles is detailed.

**2.5.1.1 Particle’s size**

Previous studies have linked particle size and wear type [19]. The following table explains the connection between the sizes and the possible wear types**:** 34

### 2.5 已有的磨损颗粒分析技术

### 2.6 已有的磨损颗粒分析的传感器

### 2.7 传统机器学习方法用于磨损颗粒分析

### 2.8 磨损颗粒的提取问题和相关硬件

### 2.9 磨损颗粒的显微照片拍摄问题

One important trend for mechanical equipment’s maintenance is the application of condition monitoring techniques. The idea is to analyze real-time data to determine the machine’s health and respond only when maintenance is actually necessary [[1, 2](#ref1)]. Currently, techniques that get focused on include vibration analysis, acoustic emission with ultrasound, oil analysis and wear debris analysis (WDA). Each one has its own advantages and constraints. It’s widely accepted that a single technique cannot meet the requirements for all situations, still, wear debris analysis is considered as one of the most effective approach by many users of condition based maintenance techniques for the following reasons:

* Wear debris or wear particles are generated as components move relative to each other. The parameters that define wear debris such as their shape, size, surface texture, etc. reflect the wear modes, wear mechanisms, and the severity associated with their generation [[3](#ref1)[, 4](#ref10)].
* The patterns of quantity and type of wear debris changes over time have strong correlation with state of machine. It is possible to predict potential failures or component’s deterioration from very early stage to avoid catastrophic accidents [[5, 6](#ref10)].

Although WDA can provide a lot of information for problem detection and fault diagnosis, the technique has not been widely used in the industry for that the debris’ morphology assessment, classification and the machine status evaluation relies heavily on expertise, which is time-consuming, costly and objective. These limitations have driven the need to automate this technique from which the industry will benefit in terms of economy and safety [[7, 8](#ref10)].

The availability of modern instrumentation provided a wide array of studies that helped identify the characteristics of wear debris and the mechanisms by which they generated [[9](#ref10)]. Approaches by observing and analyzing the morphological features of wear debris are thus promising potential online solutions [[10, 11](#ref10)]. Until now, researchers have made considerable efforts to build auto-classification system for wear debris. It is reported that various machine learning algorithm can be used to distinguish adhesive, fatigue and abrasive particles using area, perimeter and elongation parameters [[12, 13, 14](#ref15)]. These studies showed that these simple parameters are effective for certain types of wear debris. However, there are several kinds of abnormal debris larger than 20 cannot be accurately identified. Wu Tonghai team carried out extensive research on the on-line monitoring of wear debris [[15, 1](#ref20)6, 17]. The team had independently developed a set of online visualized ferrography and proposed that the area of debris coverage and their diameters in the visible area can be used as indicators for judging the state of machine. This study provided a hardware design for on-line ferrography. However, the detailed surface texture of the debris was not obtained due to small magnification.

Another problem drawing a lot of attention is the auto-segmentation of wear debris’ image. As a crucial first step for auto-classification, the accuracy of segmentation has great impact on the subsequent processes. The complicated background of the image and debris’ overlapping phenomenon makes segmentation a hard job. It’s reported in several literatures that ant-colony algorithm, gray level integrated morphological features and J-segmentation algorithm can be used to solve this problem [[7, 1](#ref20)8, 19]. The drawback of these methods is that they all require carefully pre-processing and parameter configuration, the result of segmentation will change dramatically under different situations.

To sum up, a lot of research had been done on on-line wear debris’ analysis. The recognition of cutting, spherical and oxide abrasive grains has basically been realized through shape, size and color while classification of similar debris such as severe sliding particle, fatigue laminar particle, and chunky spalling particle is not entirely successful yet. Segmentation of overlapping debris or ones that formed in chains remains unsolved.

Deep neural networks (DNNs), also referred to as deep learning, are a sub-field of machine learning which aims at giving computers the ability to learn without being explicitly programmed [20]. DNNs are currently widely used for many artificial intelligence jobs including computer vision, speech recognition, and robotics [[21, 22](#ref20)]. DNNs have broad prospects for wear debris’ detection, yet relevant research is nearly blank so far.

The objective of this study is to solve the problems described above. A deep convolutional neural network is used for wear debris’ segmentation and classification at the same time, also known as instance segmentation which combines object detection and semantic segmentation. This study focused on practical matters such as building dataset consist of various kinds of wear debris, architecture setting and testing of transfer learning procedure. State-of-art accuracy of identifying as well as localizing debris from images of large magnification is achieved.

**2.1 Data collection**

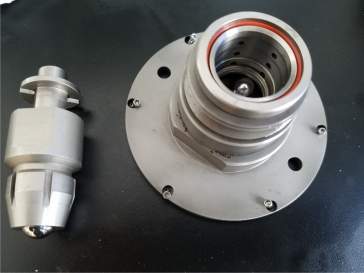
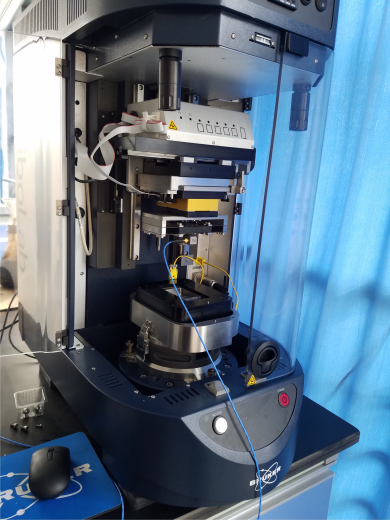
***2.1.1 Generation of wear debris***

The testing machine, showed in fig.[1](#fig2), has a universal base that can be equipped with a range of drive modules simulating rotational, linear, or oscillating motions as well as an upper carriage fitted with a force measuring sensor. Three kinds of tribology tests were performed to simulate different wear modes:

**Pin on disk** consists of a stationary pin loaded against a rotating disk. (The tests were conducted in a laboratory air with temperature of 22℃ and a relative humidity of 50%). The upper pin made of standard 416 stainless steel is cylindrical in shape with polished flat ends of 8*mm*, the disk is made of alloy steel E52100 with surface roughness RA=0.02μm. The Mobile Gard 412 was used as lubricant. The tests were conducted at 900r/m, which means the linear velocity reached 170mm/s. A load of 30kg was applied to the pin for 25h. Pin on disk test was mainly for generating adhesive wear debris. Adhesive wear particles were the main product of this test. Adhesive wear is the transfer of material from one contacting surface to another. The tests were set this long so that both mild and severe debris can be generated. Some pictures of adhesive debris are showed in first row of fig2. They are mostly small particles formed in chains, especially in last two pictures debris stick together due to magnetization which is quite common in ferrograph [[7](#ref20)].

Compared to pin on disk test, **Pin on plate test** replaced the rotating module by a reciprocating module. The upper pin is made of cast iron HT250 and the plate is made of Gr15 steel with surface roughness RA=0.2μm. The stroke was set to 18mm and frequency was set to 4Hz. The wear particles were generated with a 48kg load for 12h. Unlike pin on disk test, pin on plate test was mean to simulate the wear condition of reciprocating friction pairs such as piston-ring cylinder or hydraulic cylinder. It can be seen from the pictures in second row of fig.[2](#fig4), some of the severe sliding wear particles are quite large with clear surface scratch due to rough surface and long hours of wear process.

The fatigue wear particles were generated by **4-ball testing** machine. The material of the ball is GCr15 (hardness 63*HRC*). Maximum load and speed were set to 900N and 300r/min. In order to generate fatigue wear particles the running time was set to 50h. Fatigue begins with reduced lubrication regime and continuous stresses that exceeded the endurance limit of the material, causing cracks beneath the surface. This creates micro-pitting and eventually destructive spalling. The fatigue particles are demonstrated in the third row of fig.[2](#fig4), there are some micro spalling particles in the first two pictures and some large chunky spalling and laminar particles in last two pictures.



(a)Bruker UMT’s base (b) four-ball module and pin on disk module

Figure 1 Bruker’s UMT universal mechanical tester.

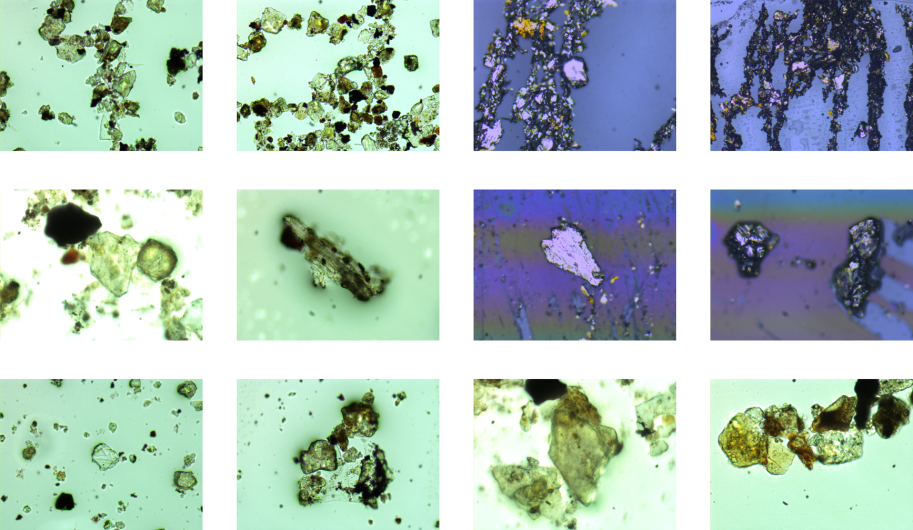


Figure 2 Examples of wear debris generated in the tests.

***2.1.2 Semi-on-line sampling***

The most unreliable part of wear debris analysis is the method used to extract wear debris from the system. In order to take microscopic pictures from a flowing fluid, the objective lens must be very close to the debris which is exactly the cause for many problems (larger the magnification, shallower the depth of focus). For example, a window or a transparent section must be designed otherwise nothing in the tube can be seen. The difficulty is that large magnification needs both inner diameter and thickness of that section to be minimized. The section would be so fragile that flow rate would be nearly zero if it won’t explode at all. As a consequence, researchers only uses 20× magnification, and even with that, restoration techniques has to be applied to enhance the quality of the part that out of focus [15].

In this study, a semi-on-line approach by Ferrography is proposed. Ferrography uses a high-gradient magnetic field to extract wear debris from a fluid sample as it flows down a specially prepared microscope substrate. Since the debris can be attached to the surface, it’s possible to alter the window section with a detachable bottom to separate the work into two steps: debris collection and image capture. One can control the time intervals of these two steps depend on the machines he/she wants to monitor. The distribution and quantity of debris overtime could still provide valuable information.

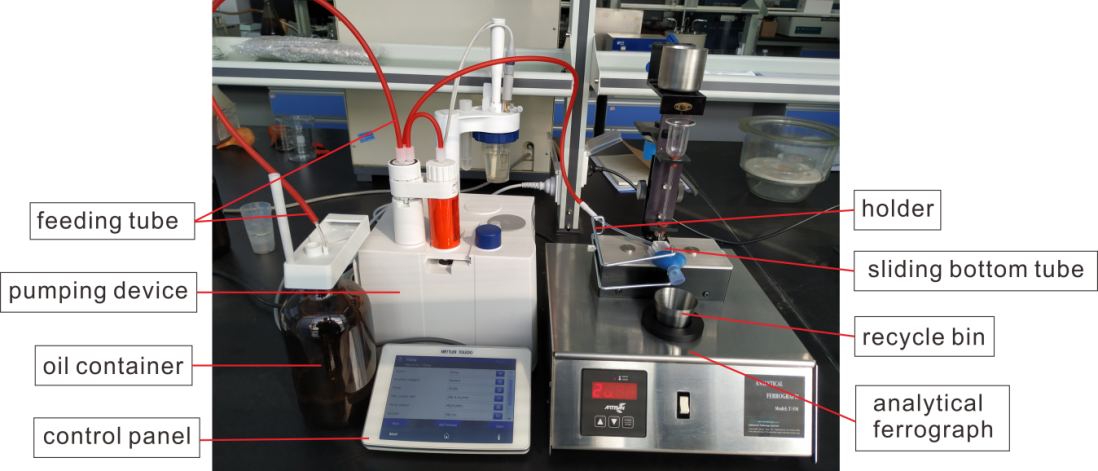


Figure 3 Connection of Analytical ferrogram and lubricating oil cycling system.

.

To simulate such semi-on-line system, lubricating oil was put in a container connecting to analytical ferrogram. The oil flow is driven by a small pump. It goes out of the container and through the tubes, finally down to an acrylic section with sliding bottom.as showed in fig.[3](#fig3). The sliding bottom has a double concave rail structure while the middle part is flat so it can hold the debris and be put under the lens. During the process, the rails are sealed with a thin layer of bending PVC as shown in fig.4. Since the Bruker’s UMT can’t attach to the system directly, lubricating oil were extracted after each kind of test and put into the container. The container, pump and tubes served as a flow system of on-line sampling process.

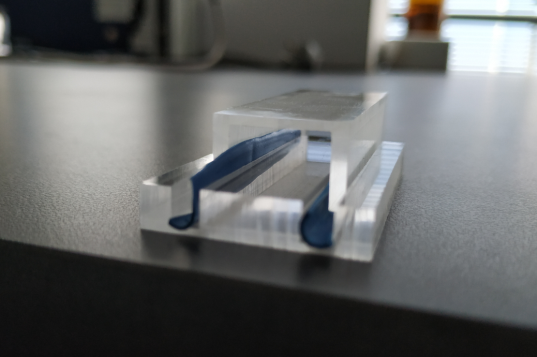
******

Figure 4 Sliding bottom with simple sealing

In this experiment, images were taken every 30 minutes of collecting. In the images capture step, the pump is stopped and the bottom detached, the residual fluid on it is washed off (another advantage of the design) before it be put under the microscope. Images were shot by optical microscopy OlypusBX51 with a Color Charge Couple Device (CCD) Camera. Most of the pictures were taken under 200× magnification because it produces sharp and clear image of debris of size 20μm to 200μm which are of particular interest in this study.

**2.2 Deep neural network**

***2.2.1 Pipe line of instance segmentation***

In this research, we focused on instance segmentation of **five kinds of wear debris** [[2,](#ref10)[1](#ref30)3]:

* Normal rubbing wear: small platelets particles less than 15μm in major dimensions.
* Cutting wear debris: curls or spirals like particles generated by abrasive wear, can be 25-100μm long and 5um thin.
* Severe sliding debris: large flaks with striation on the surface, indicating local adhesion, more than 20um in major dimension.
* Laminar rolling fatigue debris: larger than 20um in major dimension with relatively smooth surface.
* Chunky fatigue spall: chunks of rough metal, irregular shape, indicating severe fatigue spalling.

As briefly mentioned above, the instance segmentation is challenging because it requires the correct detection of all wear debris in an image while precisely segmenting each one. In this experiment, the Mask-RCNN was used to train on the dataset of wear debris described in the previous section [23]. Mask-RCNN consists of two parts: A region proposal network (RPN) to generate a set of dense candidate regions from a set of anchors containing a foreground object, i.e. wear debris; A classification network for debris’ classification, bounding box regression and prediction of segmentation mask accordingly. The main advantage of using a deep neural network is that it can learn to classify and detect object from real data. Providing better accuracy and robustness than tradition hand engineered classifier with features such as color, shape or texture property.

***2.2.2 Network Architecture***

Convolutional Neural Networks is a common form of DNNs for computer vision tasks [[24](#ref15)]. Krizhevsky et al. [[22](#ref15)] used the CNN with extended depth to obtain the best classification accuracy at the ImageNet Large Scale Visual Recognition Challenge (LSVRC) in 2012. Since then, CNN models have been dominated in all kinds of image recognition competitions. The local connections, weight sharing and pooling operations of CNNs effectively reduce the complexity of the network and therefore reduce the number of training parameters. Also the model has some degree of invariance to translation, distortion, and scaling. The networks are composed of multiple convolutional (CONV) layers as shown in fig.[5](#fig6). In such networks, each layer is supposed to generate a successively higher-level abstraction of the input data, called a feature map, which holds important unique information. Nowadays, a deep neural network can contains a thousand CONV layers [[25](#ref20)], giving the model extremely representation power. Fully-connected layers (FC) serve the purpose of final classification in the end of structure. The CONV/FC can be calculated as follows given the shape parameters in Table 1:



**O**, **I**, **W** and **B** are the matrices of the output feature maps, input feature maps, filters and biases, respectively. *U* is a given stride size. Non-linearity is referred to a non-linear activation function. Typically it is applied right after CONV or FC layer, the non-linearity is necessary for the model since the network can be simplify to a single linear function mathematically without it. The most popular non-linearity used is rectified linear unit (ReLU) because of its simplicity [[26](#ref30)]. From the demonstration figure one can see that there are also pooling and normalization layers. The definition and effect of these layers is introduced in [[27](#ref30), 34].

Table.1 Shape parameter of CONV and FC

|  |  |
| --- | --- |
| Shape Parameter | Description |
| H/W | Input feature map’s height/width |
| R/S | Filter’s height/width ( = H or W in FC) |
| E/F | Output feature map’s height/width ( = 1 in FC) |
| N | Batch size of 3-D feature maps |
| C | Number of filter/input feature map channels |
| M | Number of total filters/ Number of output feature map channels |

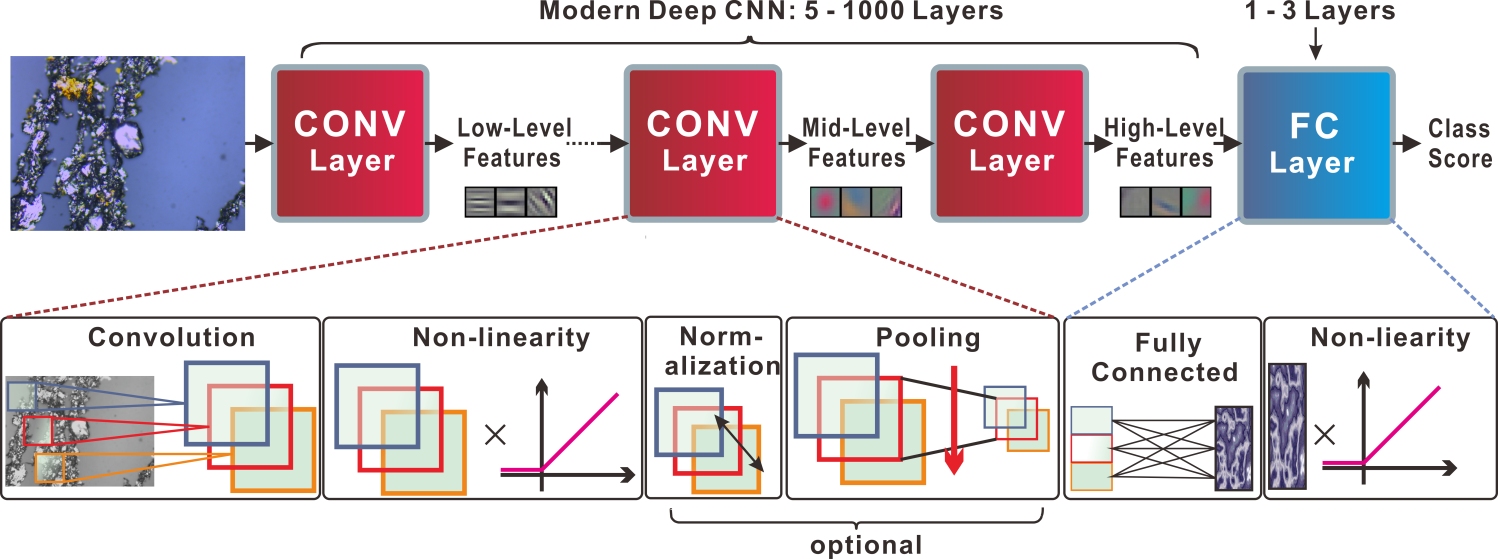
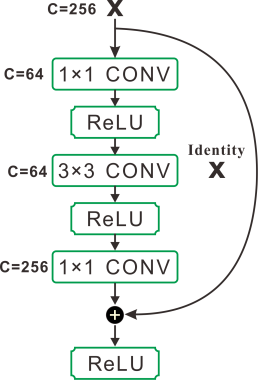
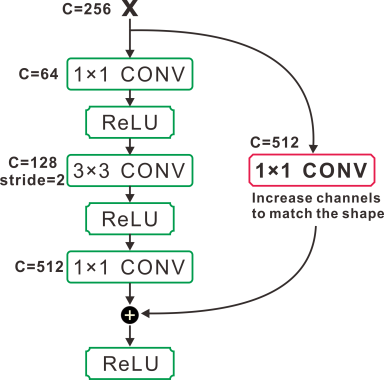


Figure 5 Framework of typical modern deep convolutional neural network.

ResNet-101 was adopted in this experiment in which each stage is composed of a number of residual blocks. As demonstrated in fig.[6](#fig9), instead of learning the function for the weight layers *F(x)*, residual block provides a shortcut that learns the residual function *F(x) = H(x)-x*). The residual connection is shown effective in very deep neural networks. The detailed layer structure of ResNet-101 is showed in table [2](#tab2). One thing to note is that in Conv phase2, 3 and 4, the first block increases dimensions as the one showed in fig.[6(b)](#fig9). The first 3×3 CONV layer and residual double the number of channels while stride is set to 2.

(a) Typical Residual block (b) Residual block that increases dimension

Figure 6 A diagram of Residual connection

Table2: Architectures of 3 CNN models. The convolutional layer parameters are denoted as ‘Receptive field, Number of channels’. The ReLU activation function and BN is not shown for brevity.

|  |  |  |
| --- | --- | --- |
| Net | Output size | Layer detail |
| Conv phase 0 | 112×112 | Conv7×7,64, stride 2 |
| Conv phase 1 | 56×56 | Maxpool3×3, stride 2 |
| Conv phase 2 | 28×28 |  |
| Conv phase 3 | 14×14 |  |
| Conv phase 4 | 7×7 |  |

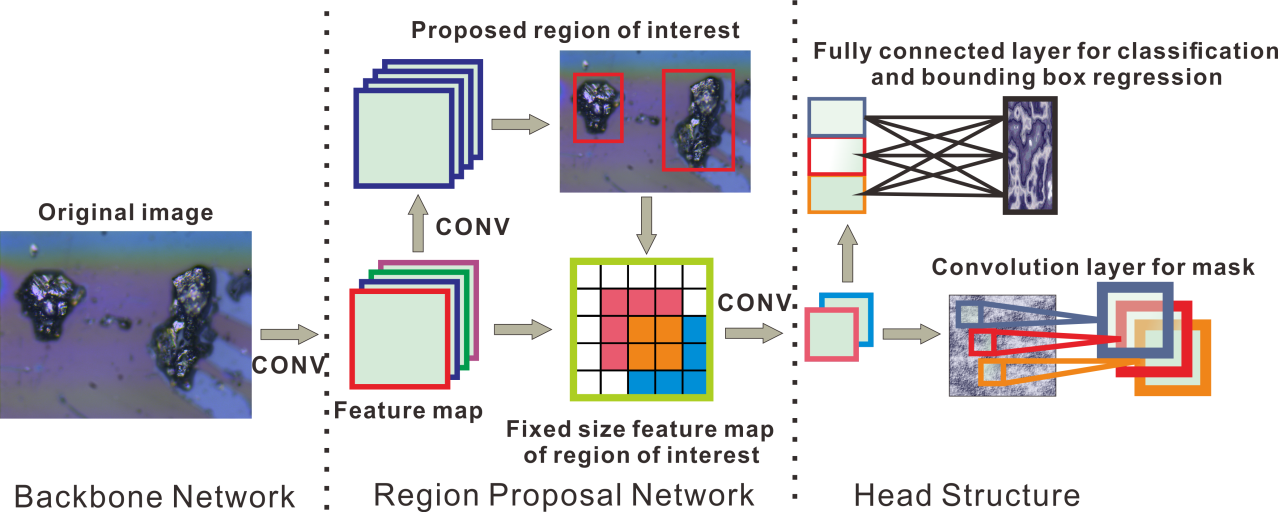
******

Figure 7 A diagram of Mask-RCNN structure

The ResNet-101’s phase one to three is the backbone feature extractor of the whole network while the phase four is the feature extractor of the final classification head. The data then flow through a FC layer for classification and bounding box regression, in parallel, the data flow through a small fully convolutional network (FCN) to generate the final mask for segmentation [28]. The RoIAlign layer is used to transform the feature map of every RoI proposed by the RPN to a fixed size feature map. Then the fixed sized feature map is fed into the head network for classification, bounding box regression and mask generation. The detailed structure is well described in [23, 29].

DNNs come in a wide variety of shapes and sizes depending

on the application. The popular shapes and sizes are also

evolving rapidly to improve accuracy and efficiency. In all

cases, the input to a DNN is a set of values representing the

information to be analyzed by the network. For instance, these

values can be pixels of an image, sampled amplitudes of an

audio wave or the numerical representation of the state of some

system or game.

The networks that process the input come in two major

forms: feed forward and recurrent as shown in Fig. 8(a). In

feed-forward networks all of the computation is performed as a

sequence of operations on the outputs of a previous layer. The

final set of operations generates the output of the network, for

example a probability that an image contains a particular object,

the probability that an audio sequence contains a particular

word, a bounding box in an image around an object or the

proposed action that should be taken. In such DNNs, the

network has no memory and the output for an input is always

the same irrespective of the sequence of inputs previously given

to the network.

In contrast, recurrent neural networks (RNNs), of which

Long Short-Term Memory networks (LSTMs) [38] are a

popular variant, have internal memory to allow long-term

dependencies to affect the output. In these networks, some

intermediate operations generate values that are stored internally

in the network and used as inputs to other operations in

conjunction with the processing of a later input. In this article,

we will focus on feed-forward networks since (1) the major

computation in RNNs is still the weighted sum, which is

covered by the feed-forward networks, and (2) to-date little

attention has been given to hardware acceleration specifically

for RNNs.

DNNs can be composed solely of fully-connected (FC)

layers (also referred to as multi-layer perceptrons, or MLP)

as shown in the leftmost layer of Fig. 8(b). In a FC layer,

all output activations are composed of a weighted sum of

all input activations (i.e., all outputs are connected to all

inputs). This requires a significant amount of storage and

computation. Thankfully, in many applications, we can remove

some connections between the activations by setting the weights

to zero without affecting accuracy. This results in a sparselyconnected

layer. A sparsely connected layer is illustrated in

the rightmost layer of Fig. 8(b).

We can also make the computation more efficient by limiting

the number of weights that contribute to an output. This sort of

structured sparsity can arise if each output is only a function

of a fixed-size window of inputs. Even further efficiency can

be gained if the same set of weights are used in the calculation

of every output. This repeated use of the same weight values is

called weight sharing and can significantly reduce the storage

requirements for weights.

An extremely popular windowed and weight-shared DNN

layer arises by structuring the computation as a convolution,

as shown in Fig. 9(a), where the weighted sum for each output

activation is computed using only a small neighborhood of input

activations (i.e., all weights beyond beyond the neighborhood

are set to zero), and where the same set of weights are shared for

every output (i.e., the filter is space invariant). Such convolutionbased

layers are referred to as convolutional (CONV) layers. 6

A. Convolutional Neural Networks (CNNs)

A common form of DNNs is Convolutional Neural Nets

(CNNs), which are composed of multiple CONV layers as

shown in Fig. 10. In such networks, each layer generates a

successively higher-level abstraction of the input data, called

a feature map (fmap), which preserves essential yet unique

information. Modern CNNs are able to achieve superior performance

by employing a very deep hierarchy of layers. CNN

are widely used in a variety of applications including image

understanding [3], speech recognition [39], game play [6],

robotics [32], etc. This paper will focus on its use in image

processing, specifically for the task of image classification [3].

Each of the CONV layers in CNN is primarily composed of

high-dimensional convolutions as shown in Fig. 9(b). In this

computation, the input activations of a layer are structured as

a set of 2-D input feature maps (ifmaps), each of which is

called a channel. Each channel is convolved with a distinct

2-D filter from the stack of filters, one for each channel; this

stack of 2-D filters is often referred to as a single 3-D filter.

The results of the convolution at each point are summed across

all the channels. In addition, a 1-D bias can be added to the

filtering results, but some recent networks [15] remove its

usage from parts of the layers. The result of this computation

is the output activations that comprise one channel of output

feature map (ofmap). Additional 3-D filters can be used on

6Note: the structured sparsity in CONV layers is orthogonal to the sparsity

that occurs from network pruning as described in Section VII-B2.

7

**R**

filter (weights)

**S**

**E**

**F**

**Partial Sum** (psum)

**Accumulation**

input fmap output fmap

**Element-wise**

**Multiplication**

**H**

**W**

**an output**

**activation**

(a) 2-D convolution in traditional image processing

**Input fmaps**

**Filters**

**Output fmaps**

**R**

**S**

**C**

**…**

**H**

**W**

**C**

**…**

**E**

**F**

**M**

**E**

**F**

**M**

**…**

**R**

**S**

**C**

**H**

**W**

**C**

**1**

**N**

**1**

**M**

**1**

**N**

(b) High dimensional convolutions in CNNs

Fig. 9. Dimensionality of convolutions.

Modern **Deep** CNN: **5 – 1000** Layers

**Class**

**Scores**

**FC**

**Layer**

**CONV**

**Layer**

**Low-Level**

**Features CONV**

**Layer**

**High-Level**

**… Features**

**1 – 3** Layers

Convolu'on Non-linearity

×

Normaliza'on Pooling

Optional

Fully

Connected

×

Non-linearity

**CONV**

**Layer**

**Mid-Level**

**Features**

Fig. 10. Convolutional Neural Networks.

the same input to create additional output channels. Finally,

multiple input feature maps may be processed together as a

batch to potentially improve reuse of the filter weights.

Given the shape parameters in Table I, the computation of

a CONV layer is defined as

O[z][u][x][y] = B[u] +

CX􀀀1

k=0

SX􀀀1

i=0

RX􀀀1

j=0

I[z][k][Ux + i][Uy + j] \_W[u][k][i][j];

0 \_ z < N; 0 \_ u < M; 0 \_ x < F; 0 \_ y < E;

E = (H 􀀀 R + U)=U; F = (W 􀀀 S + U)=U:

(1)

O, I, W and B are the matrices of the ofmaps, ifmaps, filters

and biases, respectively. U is a given stride size. Fig. 9(b)

shows a visualization of this computation (ignoring biases).

To align the terminology of CNNs with the generic DNN,

\_ filters are composed of weights (i.e., synapses)

\_ input and output feature maps (ifmaps, ofmaps) are

composed of activations (i.e., input and output neurons)

Shape Parameter Description

N batch size of 3-D fmaps

M # of 3-D filters / # of ofmap channels

C # of ifmap/filter channels

H=W ifmap plane height/width

R=S filter plane height/width (= H or W in FC)

E=F ofmap plane height/width (= 1 in FC)

TABLE I

SHAPE PARAMETERS OF A CONV/FC LAYER.

**Sigmoid**

1

-1

0

-1 0 1

**y=1/(1+e-x)**

**Hyperbolic Tangent**

1

-1

0

-1 0 1

**y=(ex-e-x)/(ex+e-x)**

**Rectified Linear Unit**

**(ReLU)**

1

-1

0

-1 0 1

**y=max(0,x)**

**Leaky ReLU**

1

-1

0

-1 0 1

**y=max(αx,x)**

**Exponential LU**

1

-1

0

-1 0 1

**x,**

**α(ex-1),**

**x≥0**

**y= x<0**

α = small const. (e.g. 0.1)

**Traditional**

**Non-Linear**

**Activation**

**Functions**

**Modern**

**Non-Linear**

**Activation**

**Functions**

Fig. 11. Various forms of non-linear activation functions (Figure adopted

from Caffe Tutorial [46]).

From five [3] to more than a thousand [15] CONV layers

are commonly used in recent CNN models. A small number,

e.g., 1 to 3, of fully-connected (FC) layers are typically applied

after the CONV layers for classification purposes. A FC layer

also applies filters on the ifmaps as in the CONV layers, but

the filters are of the same size as the ifmaps. Therefore, it

does not have the weight sharing property of CONV layers.

Eq. (1) still holds for the computation of FC layers with a

few additional constraints on the shape parameters: H = R,

F = S, E = F = 1, and U = 1.

In addition to CONV and FC layers, various optional layers

can be found in a DNN such as the non-linearity, pooling,

and normalization. The function and computations for each of

these layers are discussed next.

1) Non-Linearity: A non-linear activation function is typically

applied after each CONV or FC layer. Various non-linear

functions are used to introduce non-linearity into the DNN as

shown in Fig. 11. These include historically conventional nonlinear

functions such as sigmoid or hyperbolic tangent as well

as rectified linear unit (ReLU) [40], which has become popular

in recent years due to its simplicity and its ability to enable

fast training. Variations of ReLU, such as leaky ReLU [41],

parametric ReLU [42], and exponential LU [43] have also been

explored for improved accuracy. Finally, a non-linearity called

maxout, which takes the max value of two intersecting linear

functions, has shown to be effective in speech recognition

tasks [44, 45].

2) Pooling: A variety of computations that reduce the

dimensionality of a feature map are referred to as pooling.

Pooling, which is applied to each channel separately, enables

8

9 3 5 3

10 32 2 2

1 3 21 9

2 6 11 7

2x2 pooling, stride 2

32 5

6 21

**Max** pooling **Average** pooling

18 3

3 12

Fig. 12. Various forms of pooling (Figure adopted from Caffe Tutorial [46]).

the network to be robust and invariant to small shifts and

distortions. Pooling combines, or pools, a set of values in

its receptive field into a smaller number of values. It can be

configured based on the size of its receptive field (e.g., 2\_2)

and pooling operation (e.g., max or average), as shown in

Fig. 12. Typically pooling occurs on non-overlapping blocks

(i.e., the stride is equal to the size of the pooling). Usually a

stride of greater than one is used such that there is a reduction

in the dimension of the representation (i.e., feature map).

3) Normalization: Controlling the input distribution across

layers can help to significantly speed up training and improve

accuracy. Accordingly, the distribution of the layer input

activations (\_, \_) are normalized such that it has a zero mean

and a unit standard deviation. In batch normalization (BN),

the normalized value is further scaled and shifted, as shown

in Eq. (2), where the parameters (, \_) are learned from

training [47]. \_ is a small constant to avoid numerical problems.

Prior to this, local response normalization (LRN) [3] was

used, which was inspired by lateral inhibition in neurobiology

where excited neurons (i.e., high value activations) should

subdue its neighbors (i.e., cause low value activations); however,

BN is now considered standard practice in the design of

CNNs while LRN is mostly deprecated. Note that while LRN

usually is performed after the non-linear function, BN is mostly

performed between the CONV or FC layer and the non-linear

function.

y =

x 􀀀 \_

p

\_2 + \_

+ \_ (2)

B. Popular DNN Models

Many DNN models have been developed over the past

two decades. Each of these models has a different ‘network

architecture’ in terms of number of layers, layer types, layer

shapes (i.e., filter size, number of channels and filters), and

connections between layers. Understanding these variations

and trends is important for incorporating the right flexibility

in any efficient DNN engine.

In this section, we will give an overview of various popular

DNNs such as LeNet [48] as well as those that competed in

and/or won the ImageNet Challenge [14] as shown in Fig. 7,

most of whose models with pre-trained weights are publicly

available for download; the DNN models are summarized in

Table II. Two results for top-5 error results are reported. In the

first row, the accuracy is boosted by using multiple crops from

the image and an ensemble of multiple trained models (i.e.,

the DNN needs to be run several times); these results were

used to compete in the ImageNet Challenge. The second row

reports the accuracy if only a single crop was used (i.e., the

DNN is run only once), which is more consistent with what

would likely be deployed in real-time and/or energy-constrained

applications.

LeNet [11] was one of the first CNN approaches introduced

in 1989. It was designed for the task of digit classification in

grayscale images of size 28\_28. The most well known version,

LeNet-5, contains two CONV layers and two FC layers [48].

Each CONV layer uses filters of size 5\_5 (1 channel per filter)

with 6 filters in the first layer and 16 filters in the second layer.

Average pooling of 2\_2 is used after each convolution and a

sigmoid is used for the non-linearity. In total, LeNet requires

60k weights and 341k multiply-and-accumulates (MACs) per

image. LeNet led to CNNs’ first commercial success, as it was

deployed in ATMs to recognize digits for check deposits.

AlexNet [3] was the first CNN to win the ImageNet Challenge

in 2012. It consists of five CONV layers and three FC layers.

Within each CONV layer, there are 96 to 384 filters and the

filter size ranges from 3\_3 to 11\_11, with 3 to 256 channels

each. In the first layer, the 3 channels of the filter correspond

to the red, green and blue components of the input image.

A ReLU non-linearity is used in each layer. Max pooling of

3\_3 is applied to the outputs of layers 1, 2 and 5. To reduce

computation, a stride of 4 is used at the first layer of the

network. AlexNet introduced the use of LRN in layers 1 and

2 before the max pooling, though LRN is no longer popular

in later CNN models. One important factor that differentiates

AlexNet from LeNet is that the number of weights is much

larger and the shapes vary from layer to layer. To reduce the

amount of weights and computation in the second CONV layer,

the 96 output channels of the first layer are split into two groups

of 48 input channels for the second layer, such that the filters in

the second layer only have 48 channels. Similarly, the weights

in fourth and fifth layer are also split into two groups. In total,

AlexNet requires 61M weights and 724M MACs to process

one 227\_227 input image.

Overfeat [49] has a very similar architecture to AlexNet with

five CONV layers and three FC layers. The main differences

are that the number of filters is increased for layers 3 (384

to 512), 4 (384 to 1024), and 5 (256 to 1024), layer 2 is not

split into two groups, the first fully connected layer only has

3072 channels rather than 4096, and the input size is 231\_231

rather than 227\_227. As a result, the number of weights grows

to 146M and the number of MACs grows to 2.8G per image.

Overfeat has two different models: fast (described here) and

accurate. The accurate model used in the ImageNet Challenge

gives a 0.65% lower top-5 error rate than the fast model at the

cost of 1.9\_ more MACs

VGG-16 [50] goes deeper to 16 layers consisting of 13

CONV layers and 3 FC layers. In order to balance out the

cost of going deeper, larger filters (e.g., 5\_5) are built from

multiple smaller filters (e.g., 3\_3), which have fewer weights,

to achieve the same receptive fields as shown in Fig. 13(a).

As a result, all CONV layers have the same filter size of 3\_3.

In total, VGG-16 requires 138M weights and 15.5G MACs

to process one 224\_224 input image. VGG has two different

models: VGG-16 (described here) and VGG-19. VGG-19 gives

a 0.1% lower top-5 error rate than VGG-16 at the cost of

1.27\_ more MACs.

9

5x5 filter Two 3x3 filters

decompose

Apply sequentially

(a) Constructing a 5\_5 support from 3\_3 filters. Used in VGG-16.

decompose

5x5 filter 5x1 filter

1x5 filter

Apply sequentially

(b) Constructing a 5\_5 support from 1\_5 and 5\_1 filter. Used in

GoogleNet/Inception v3 and v4.

Fig. 13. Decomposing larger filters into smaller filters.

GoogLeNet [51] goes even deeper with 22 layers. It introduced

an inception module, shown in Fig. 14, which is

composed of parallel connections, whereas previously there

was only a single serial connection. Different sized filters (i.e.,

1\_1, 3\_3, 5\_5), along with 3\_3 max-pooling, are used for

each parallel connection and their outputs are concatenated

for the module output. Using multiple filter sizes has the

effect of processing the input at multiple scales. For improved

training speed, GoogLeNet is designed such that the weights

and the activations, which are stored for backpropagation during

training, could all fit into the GPU memory. In order to reduce

the number of weights, 1\_1 filters are applied as a ‘bottleneck’

to reduce the number of channels for each filter [52]. The 22

layers consist of three CONV layers, followed by 9 inceptions

layers (each of which are two CONV layers deep), and one FC

layer. Since its introduction in 2014, GoogleNet (also referred

to as Inception) has multiple versions: v1 (described here), v3 7

and v4. Inception-v3 decomposes the convolutions by using

smaller 1-D filters as shown in Fig. 13(b) to reduce number

of MACs and weights in order to go deeper to 42 layers.

In conjunction with batch normalization [47], v3 achieves

over 3% lower top-5 error than v1 with 2.5\_ increase in

computation [53]. Inception-v4 uses residual connections [54],

described in the next section, for a 0.4% reduction in error.

ResNet [15], also known as Residual Net, uses residual

connections to go even deeper (34 layers or more). It was

the first entry DNN in ImageNet Challenge that exceeded

human-level accuracy with a top-5 error rate below 5%. One

of the challenges with deep networks is the vanishing gradient

during training: as the error backpropagates through the network

the gradient shrinks, which affects the ability to update the

weights in the earlier layers for very deep networks. Residual

net introduces a ‘shortcut’ module which contains an identity

connection such that the weight layers (i.e., CONV layers)

can be skipped as shown in Fig. 15. Rather than learning the

function for the weight layers F(x), the shortcut module learns

the residual mapping (F(x) = H(x) 􀀀 x). Initially, F(x) is

zero and the identity connection is taken; then gradually during

training, the actual forward connection through the weight layer

7v2 is very similar to v3.

1x1 CONV 3x3 CONV 5x5 CONV 1x1 CONV

1x1 CONV 1x1 CONV 3x3 MAX POOL

Input

feature

map

Output

feature

map

C=64 C=192

C=64 C=128 C=32 C=32

C=192

C=64

C=256

Fig. 14. Inception module from GoogleNet [51] with example channel lengths.

Note that each CONV layer is followed by a ReLU (not drawn).

3x3 CONV

ReLU

ReLU

3x3 CONV

+

x

F(x)

H(x) = F(x) + x

**Iden%ty**

x

(a) Without bottleneck

3x3 CONV

ReLU

ReLU

1x1 CONV

+

x

F(x)

H(x) = F(x) + x

1x1 CONV

ReLU

**Iden%ty**

x

(b) With bottleneck

Fig. 15. Shortcut module from ResNet [15]. Note that ReLU following last

CONV layer in short cut is after the addition.

is used. This is similar to the LSTM networks that are used for

sequential data. ResNet also uses the ‘bottleneck’ approach of

using 1\_1 filters to reduce the number of weight parameters.

As a result, the two layers in the shortcut module are replaced

by three layers (1\_1, 3\_3, 1\_1) where the 1\_1 reduces and

then increases (restores) the number of weights. ResNet-50

consists of one CONV layer, followed by 16 shortcut layers

(each of which are three CONV layers deep), and one FC

layer; it requires 25.5M weights and 3.9G MACs per image.

There are various versions of ResNet with multiple depths

(e.g., without bottleneck: 18, 34; with bottleneck: 50, 101, 152).

The ResNet with 152 layers was the winner of the ImageNet

Challenge requiring 11.3G MACs and 60M weights. Compared

to ResNet-50, it reduces the top-5 error by around 1% at the

cost of 2.9\_ more MACs and 2.5\_ more weights.

Several trends can be observed in the popular DNNs shown

in Table II. Increasing the depth of the network tends to provide

higher accuracy. Controlling for number of weights, a deeper

network can support a wider range of non-linear functions

10

that are more discriminative and also provides more levels

of hierarchy in the learned representation [15, 50, 51, 55].

The number of filter shapes continues to vary across layers,

thus flexibility is still important. Furthermore, most of the

computation has been placed on CONV layers rather than FC

layers. In addition, the number of weights in the FC layers is

reduced and in most recent networks (since GoogLeNet) the

CONV layers also dominate in terms of weights. Thus, the

focus of hardware implementations should be on addressing

the efficiency of the CONV layers, which in many domains

are increasingly important.

IV. DNN DEVELOPMENT RESOURCES

One of the key factors that has enabled the rapid development

of DNNs is the set of development resources that have been

made available by the research community and industry. These

resources are also key to the development of DNN accelerators

by providing characterizations of the workloads and facilitating

the exploration of trade-offs in model complexity and accuracy.

This section will describe these resources such that those who

are interested in this field can quickly get started.

A. Frameworks

For ease of DNN development and to enable sharing of

trained networks, several deep learning frameworks have been

developed from various sources. These open source libraries

contain software libraries for DNNs. Caffe was made available

in 2014 from UC Berkeley [46]. It supports C, C++, Python

and MATLAB. Tensorflow was released by Google in 2015,

and supports C++ and python; it also supports multiple CPUs

and GPUs and has more flexibility than Caffe, with the

computation expressed as dataflow graphs to manage the

tensors (multidimensional arrays). Another popular framework

is Torch, which was developed by Facebook and NYU and

supports C, C++ and Lua. There are several other frameworks

such as Theano, MXNet, CNTK, which are described in [60].

There are also higher-level libraries that can run on top of

the aforementioned frameworks to provide a more universal

experience and faster development. One example of such

libraries is Keras, which is written in Python and supports

Tensorflow, CNTK and Theano.

The existence of such frameworks are not only a convenient

aid for DNN researchers and application designers, but they

are also invaluable for engineering high performance or more

efficient DNN computation engines. In particular, because the

frameworks make heavy use of a set primitive operations,

such processing of a CONV layer, they can incorporate use of

optimized software or hardware accelerators. This acceleration

is transparent to the user of the framework. Thus, for example,

most frameworks can use Nvidia’s cuDNN library for rapid

execution on Nvidia GPUs. Similarly, transparent incorporation

of dedicated hardware accelerators can be achieved as was

done with the Eyeriss chip [61].

Finally, these frameworks are a valuable source of workloads

for hardware researchers. They can be used to drive experimental

designs for different workloads, for profiling different

workloads and for exploring hardware-software trade-offs.

B. Models

Pretrained DNN models can be downloaded from various

websites [56–59] for the various different frameworks. It should

be noted that even for the same DNN (e.g., AlexNet) the

accuracy of these models can vary by around 1% to 2%

depending on how the model was trained, and thus the results

do not always exactly match the original publication.

C. Popular Datasets for Classification

It is important to factor in the difficulty of the task when

comparing different DNN models. For instance, the task of

classifying handwritten digits from the MNIST dataset [62]

is much simpler than classifying an object into one of 1000

classes as is required for the ImageNet dataset [14](Fig. 16).

It is expected that the size of the DNNs (i.e., number of

weights) and the number of MACs will be larger for the more

difficult task than the simpler task and thus require more

energy and have lower throughput. For instance, LeNet-5[48]

is designed for digit classification, while AlexNet[3], VGG-

16[50], GoogLeNet[51], and ResNet[15] are designed for the

1000-class image classification.

There are many AI tasks that come with publicly available

datasets in order to evaluate the accuracy of a given DNN.

Public datasets are important for comparing the accuracy of

different approaches. The simplest and most common task

is image classification, which involves being given an entire

image, and selecting 1 of N classes that the image most likely

belongs to. There is no localization or detection.

MNIST is a widely used dataset for digit classification

that was introduced in 1998 [62]. It consists of 28\_28 pixel

grayscale images of handwritten digits. There are 10 classes

(for 10 digits) and 60,000 training images and 10,000 test

images. LeNet-5 was able to achieve an accuracy of 99.05%

when MNIST was first introduced. Since then the accuracy has

increased to 99.79% using regularization of neural networks

with dropconnect [63]. Thus, MNIST is now considered a fairly

easy dataset.

CIFAR is a dataset that consists of 32\_32 pixel colored

images of of various objects, which was released in 2009 [64].

CIFAR is a subset of the 80 million Tiny Image dataset [65].

CIFAR-10 is composed of 10 mutually exclusive classes. There

are 50,000 training images (5000 per class) and 10,000 test

images (1000 per class). A two-layer convolutional deep belief

network was able to achieve 64.84% accuracy on CIFAR-10

when it was first introduced [66]. Since then the accuracy has

increased to 96.53% using fractional max pooling [67].

ImageNet is a large scale image dataset that was first

introduced in 2010; the dataset stabilized in 2012 [14]. It

contains images of 256\_256 pixel in color with 1000 classes.

The classes are defined using the WordNet as a backbone to

handle ambiguous word meanings and to combine together

synonyms into the same object category. In otherwords, there

is a hierarchy for the ImageNet categories. The 1000 classes

were selected such that there is no overlap in the ImageNet

hierarchy. The ImageNet dataset contains many fine-grained

categories including 120 different breeds of dogs. There are

1.3M training images (732 to 1300 per class), 100,000 testing

11

Metrics LeNet AlexNet Overfeat VGG GoogLeNet ResNet

5 fast 16 v1 50

Top-5 errory n/a 16.4 14.2 7.4 6.7 5.3

Top-5 error (single crop)y n/a 19.8 17.0 8.8 10.7 7.0

Input Size 28\_28 227\_227 231\_231 224\_224 224\_224 224\_224

# of CONV Layers 2 5 5 13 57 53

Depth in # of CONV Layers 2 5 5 13 21 49

Filter Sizes 5 3,5,11 3,5,11 3 1,3,5,7 1,3,7

# of Channels 1, 20 3-256 3-1024 3-512 3-832 3-2048

# of Filters 20, 50 96-384 96-1024 64-512 16-384 64-2048

Stride 1 1,4 1,4 1 1,2 1,2

Weights 2.6k 2.3M 16M 14.7M 6.0M 23.5M

MACs 283k 666M 2.67G 15.3G 1.43G 3.86G

# of FC Layers 2 3 3 3 1 1

Filter Sizes 1,4 1,6 1,6,12 1,7 1 1

# of Channels 50, 500 256-4096 1024-4096 512-4096 1024 2048

# of Filters 10, 500 1000-4096 1000-4096 1000-4096 1000 1000

Weights 58k 58.6M 130M 124M 1M 2M

MACs 58k 58.6M 130M 124M 1M 2M

Total Weights 60k 61M 146M 138M 7M 25.5M

Total MACs 341k 724M 2.8G 15.5G 1.43G 3.9G

Pretrained Model Website [56]z [57, 58] n/a [57–59] [57–59] [57–59]

TABLE II

SUMMARY OF POPULAR DNNS [3, 15, 48, 50, 51]. yACCURACY IS MEASURED BASED ON TOP-5 ERROR ON IMAGENET [14]. zTHIS VERSION OF LENET-5

HAS 431K WEIGHTS FOR THE FILTERS AND REQUIRES 2.3M MACS PER IMAGE, AND USES RELU RATHER THAN SIGMOID.

**MNIST ImageNet**

Fig. 16. MNIST (10 classes, 60k training, 10k testing) [62] vs. ImageNet

(1000 classes, 1.3M training, 100k testing)[14] dataset.

images (100 per class) and 50,000 validation images (50 per

class).

The accuracy of the ImageNet Challenge are reported using

two metrics: Top-5 and Top-1 error. Top-5 error means that if

any of the top five scoring categories are the correct category,

it is counted as a correct classification. The Top-1 requires

that the top scoring category be correct. In 2012, the winner

of the ImageNet Challenge (AlexNet) was able to achieve an

accuracy of 83.6% for the top-5 (which is substantially better

than the 73.8% which was second place that year that did not

use DNNs); it achieved 61.9% on the top-1 of the validation

set. In 2017, the highest accuracy was 97.7% for the top-5.

In summary of the various image classification datasets, it

is clear that MNIST is a fairly easy dataset, while ImageNet

is a challenging one with a wider coverage of classes. Thus

in terms of evaluating the accuracy of a given DNN, it is

important to consider that dataset upon which the accuracy is

measured.

D. Datasets for Other Tasks

Since the accuracy of the state-of-the-art DNNs are performing

better than human-level accuracy on image classification

tasks, the ImageNet Challenge has started to focus on more

difficult tasks such as single-object localization and object

detection. For single-object localization, the target object must

be localized and classified (out of 1000 classes). The DNN

outputs the top five categories and top five bounding box

locations. There is no penalty for identifying an object that

is in the image but not included in the ground truth. For

object detection, all objects in the image must be localized

and classified (out of 200 classes). The bounding box for all

objects in these categories must be labeled. Objects that are

not labeled are penalized as are duplicated detections.

Beyond ImageNet, there are also other popular image

datasets for computer vision tasks. For object detection, there

is the PASCAL VOC (2005-2012) dataset that contains 11k

images representing 20 classes (27k object instances, 7k of

which has detailed segmentation) [68]. For object detection,

segmentation and recognition in context, there is the MS COCO

dataset with 2.5M labeled instances in 328k images (91 object

categories) [69]; compared to ImageNet, COCO has fewer

categories but more instances per category, which is useful for

precise 2-D localization. COCO also has more labeled instances

per image to potentially help with contextual information.

Most recently even larger scale datasets have been made

available. For instance, Google has an Open Images dataset

with over 9M images [70], spanning 6000 categories. There is

also a YouTube dataset with 8M videos (0.5M hours of video)

covering 4800 classes [71]. Google also released an audio

dataset comprised of 632 audio event classes and a collection

of 2M human-labeled 10-second sound clips [72]. These large

datasets will be evermore important as DNNs become deeper

with more weight parameters to train.

Undoubtedly, both larger datasets and datasets for new

domains will serve as important resources for profiling and

exploring the efficiency of future DNN engines.

### 2.10 磨损颗粒图像的总体问题概述

### 2.11 基于文献的理性讨论

### 2.12 本章结论

## Chapter3 研究方法/思路/方向介绍

（1）通过一系列试验建立适用于磨损颗粒图像高速检测和识别的深度神经网络结构，为磨损颗粒自动识别提炼出有效模型。

（2）优化网络的训练方法，研究先验知识对识别的影响，完成准确分割并识别疲劳块状磨粒、层状颗粒、严重滑动磨粒以及滑动滚动混合颗粒的任务，为铁谱在线化应用提供技术支持。

（3）通过特征可视化技术总结深度神经网络在磨损颗粒特征提取中的基本规律，为纹理图像分类问题提供依据。

**2.3 拟解决关键问题**

（1）**从背景中实时准确地分割出所有单个磨粒是一个图像分割难题**。因显微图像背景复杂，颗粒边界模糊，导致传统的分水岭算法、阈值分割、边界算子等方法时常导致欠分割或过度分割。在运用深度神经网络解决分割问题时，先验知识输入、模块的连接方法对分割的准确性会有显著的影响，为此需要进行大量的试验。为了达到实时的检测速度，整体的结构和相应的训练策略又需要最大程度的优化，否则对机械设备在线监测的意义就会大大降低，**同时保证精度和速度是本次研究面临的最大挑战**。

（2）**实现疲劳块状磨粒、层状颗粒以及严重滑动磨粒等非正常磨损大颗粒的精确识别是本项目的关键问题。**与主流的图像识别任务不同，磨损颗粒的分类并无可以参考的案例，所有参数都需要从零开始测试分析，如何运用卷积神经网络提取以纹理特征为中心的特征向量，通过何种预训练方法和正则化方法能保证深度网络不会发生过拟合且保证识别准确率，如何设置学习率和误差函数，**这类细节对于缺乏完善理论支持的深度神经网络既是技术细节，也是重要科学问题**，会极大影响识别的准确率。

**3.1 研究方案**

根据以上研究内容，结合本项目的研究目标，本项目的整体技术路线如图3所示。本项目的研究方案包括：

结合已有的磨损颗粒图像样本，加上实验室严格控制温度、荷载、转速条件下的摩擦磨损试验机，以及不同工况下的船用发动机、齿轮箱及轴承台架试验采集不同磨损机理、不同磨损烈度下的磨粒图像，使用人工标记颗粒类型和边界后的图片作为训练和测试样本，开展对深度神经网络总体结构模型设计、先验知识对分割质量的影响以及神经网络的训练方法、正则化方法等关键问题的全面深入研究，最终实现磨粒即时且准确的智能识别。其中，人工对图像中颗粒边界标注的工作量较大，是前期的基础工作；而深度网络的结构设计、模块和连接方式研究、训练和正则化方法等细节则需要反复的交叉验证，则是本项目的研究核心

**（1）铁谱磨粒图库的建立**

本研究需要建立在严格控制下的样本采集基础上，通过实验室严格控制温度、荷载、转速条件下的摩擦磨损试验机，制造不同磨损机理下不同磨损剧烈程度下的各类磨粒，如图4所示。其中，销盘磨损机主要用于生成粘着磨损颗粒，荷载分别为150N，300N，450N,对应转速为900rpm，其中高档位的设计是为了生成严重滑动磨损颗粒，转头和转盘材料拟分别使用不锈钢和低碳钢，经过一系列实验，这种参数设定不容易产生局部咬死。疲劳颗粒的生成则用4球磨损测试仪，最大荷载和转速分别设定为1800N和300rpm。混合型颗粒则用标准齿轮测试机生成，齿轮机在不同荷载下，分别运转4个20分钟度过咬合期，在此基础上将齿轮机在较高荷载下连续工作20h，每两小时取样本，收集不同时段的疲劳颗粒。所有机械开机时保持实验室相对湿度50%，温度22°左右。

另一方面，进行船用发动机、齿轮箱及轴承多工况台架试验，除正常工况，另外通过减少进油量，增大负荷以模拟恶劣润滑工况下的摩擦副运转，收集气缸油、齿轮箱油以及轴承润滑系统油中的磨损颗粒。

收集好颗粒样本后，用直读铁谱仪进行制谱，光学显微镜进行图像拍摄，并**人工进行分类以及边界标注**（此步骤工作量极大）以建立详实、可靠完善的磨损颗粒图像数据库，数据库可以通过**磨损机理**和**磨损剧烈程度**两个标签共进行查找。其中磨损机理分为粘着磨损、疲劳磨损、混合磨损三大类，磨损剧烈程度则分为：正常，边界润滑，严重磨损三大类，总共研究的磨损颗粒类型为9个小类。**主要研究对象则是20μm以上的非正常大型磨损颗粒，包括疲劳块状磨粒、层状颗粒、严重滑动磨粒以及滚动滑动混合型颗粒因为这类颗粒相似度很高，是自动识别的难点。**

1. **深度神经网络结构研究**

深度神经网络的结构研究是本项目的核心研究，一个适用于铁谱磨粒图像的深度神经网络需要具有以下3个特征：

· 显微图像背景复杂，颗粒边界模糊等情况的分割效果较好；

* 对纹理特征提取能力强，可实现疲劳块状磨粒、混合磨损机理颗粒以及严重滑动磨粒等相似颗粒的准确识别；
* 保证检测速度，即时识别是磨损颗粒在线化检测的必要条件。

为了得到符合上述特点的网络模型，具体研究方案如下：

1. 确立骨架，以Mask-RCNN的结构作为骨架模板，整体结构应分为输入层、卷积层、全连接层和输出层，其中输入层为原始图像，**卷积层则兼具磨粒分类和图像分割两方面的特征提取，为最重要组成部分，基本框架结构如图5所示**，全连接层用于卷积层和输出层的连接，输出层则需设置不同的目标函数并能**同时输出**分割后的结果和图像中各个磨损颗粒的类型；
2. 以卷积层的模块作为基本研究对象，开展模块的结构设计、模块间的连接方式的**交叉验证**。模块结构的实验对象包括残差神经网络模块、分形神经网络分形块、宽度神经网络的宽度，以及对应模块中卷积层滤波器的设计和池化层尺寸试验，三种模块的结构如图6所示。通过特征可视化技术研究不同网络结构和相应的连接方式和数据流传递理论；
3. 测试传统算法中的分割特征图，如JSEG中的J-image，彩色图像梯度算子图等作为先验知识对神经网络输出精度的影响，通过插入特征图到卷积层中不同层作为输入，验证先验知识的有效性，**主要针对提升分割效果**；
4. 测试不同模块时，网络深度对分割以及分类效果的影响，其中分割效果与人工标记的边界做比较，分类效果通过分类准确率作为指标，寻找最适合磨粒图像的深度；
5. 根据分割和分类结果，重复步骤-进行循环交叉验证，达到**局部最佳优化**。

**3）通过特征可视化对神经网络模型的实现细节研究。**

如前所述，**深度神经网络目前缺乏完善的理论支持**，仍处于发展阶段，通过特征可视化技术可以窥探神经网络的实现原理并反向优化网络以适应具体问题。**对于深度神经网络而言，参数初始化方法，训练方法以及正则化方法的选择不仅是技术细节，也是最重要的科学问题，**具体研究路线如下：

1. 网络中参数的初始化和网络的预训练对后续深度网络的表现至关重要，对卷积层的每一层进行自动编码处理后，从数据库中每种类型的磨损颗粒选取最具代表性的500张作为预训练样本进行网络参数的微调。
2. 对不同的初始化策略和学习率进行分析，对输入层，中间层和全连接层进行特征可视化分析并改善特征提取的质量，保证在卷积层中，逐渐提取出诸如裂纹、坑洞、划痕等磨损颗粒表面纹理特质。
3. 规则化策略的研究。对三种规则化方法batch normalization、 weight decay以及dropout等的有效性进行对比试验，并对每一种策略中的参数进行交叉验证评估，寻找最佳参数设定，防止过拟合的发生。
4. 重复步骤-，寻找局部最优化的策略以及对应的最佳参数，保证整个网络拥有强大的纹理特征表达能力的同时，不发生严重的过拟合。

磨损颗粒智能识别是目前国内外机械设备在线状态监测的研究热点，长久以来虽然开展了基于图像工程领域和模式识别方法铁谱磨粒分类研究，为磨粒智能化识别奠定了一定理论基础，但未形成基于大量样本的，针对磨粒图像特点的分割算法的和大尺寸非正常磨损颗粒识别方法的系统性研究。而深度学习发展迅速的现如今，关于深度学习应用于这一工程问题的研究几乎为空白。本项目不仅探索深度神经网络中的最新研究成果应用于磨损颗粒图像快速检测识别的方法，还进行大量系统性的试验、交叉验证。而对大量样本进行标记的工作能为以后基于数据驱动的人工智能应用于机械设备故障诊断打下基础。因此是一项开拓性前沿性研究工作，研究内容具有鲜明的创新性主要表现在：

（1）首次引入深度神经网络对磨粒的彩色图像进行实时分割和分类，对神经网络的结构展开了深入研究的同时，对先验知识的有效性进行探索，**并对磨损颗粒的分类精度和检测速度同时提出标准**。

（2）提出用特征可视化技术反向解释深度模型的实现原理，针对磨损颗粒表面的坑洞、裂纹、划痕等具体特征微调神经网络的参数和训练策略，探索深度神经网络在纹理表达方面的细节，为将来所有纹理分类问题建立一个深度神经网络预训练模型，让以后的学者可以使用该模型轻松进行迁移学习。

（3）建立足够支持深度学习网络训练的数据库（预计原始图像样本达到6000张），在信息时代，磨损颗粒的信息非常不完善，数据库的建立除了能满足本项目的研究目标外，还能对磨损颗粒和磨损机理未来的研究提供数据，同时能金属纹理分类、金属检测等人工智能提供新的框架。

## Chapter4 实验结果以及验证

## Chapter5 结果讨论

1. 尽管有摩擦力，作者的筷子用的依然不怎么样。 [↑](#footnote-ref-1)