**一维转二维深度学习研究**

[**一** **数据集详情和数据预处理** 2](#_Toc212044141)

[1. **数据集介绍** 2](#_Toc212044142)

[2. **数据集划分** 4](#_Toc212044143)

[3. **数据预处理算法** 5](#_Toc212044144)

[4. **通用图像处理参数** 9](#_Toc212044145)

[**5.** **算法生成图像展示** 9](#_Toc212044146)

[**6.** **生成图像信息比对** 10](#_Toc212044147)

[**二** **模型选型及训练策略** 10](#_Toc212044148)

[1. **训练环境** 11](#_Toc212044149)

[2. **批次大小** 12](#_Toc212044150)

[3. **训练轮次（epochs）** 12](#_Toc212044151)

[4. **训练策略** 12](#_Toc212044152)

[**三** **实验总结** 12](#_Toc212044153)

[**1.** **准确率统计结果** 13](#_Toc212044154)

[**2.** **模型推理速度** 13](#_Toc212044155)

[**3.** **实验不足点** 13](#_Toc212044156)

[**4.** **图像算法生成的图像难以看懂** 13](#_Toc212044157)

[**四** **一维数据转二维图像进行深度学习的优点** 14](#_Toc212044158)

[1. **利用二维卷积神经网络的优势** 14](#_Toc212044159)

[2. **更好地发现隐藏的关联性/结构信息** 14](#_Toc212044160)

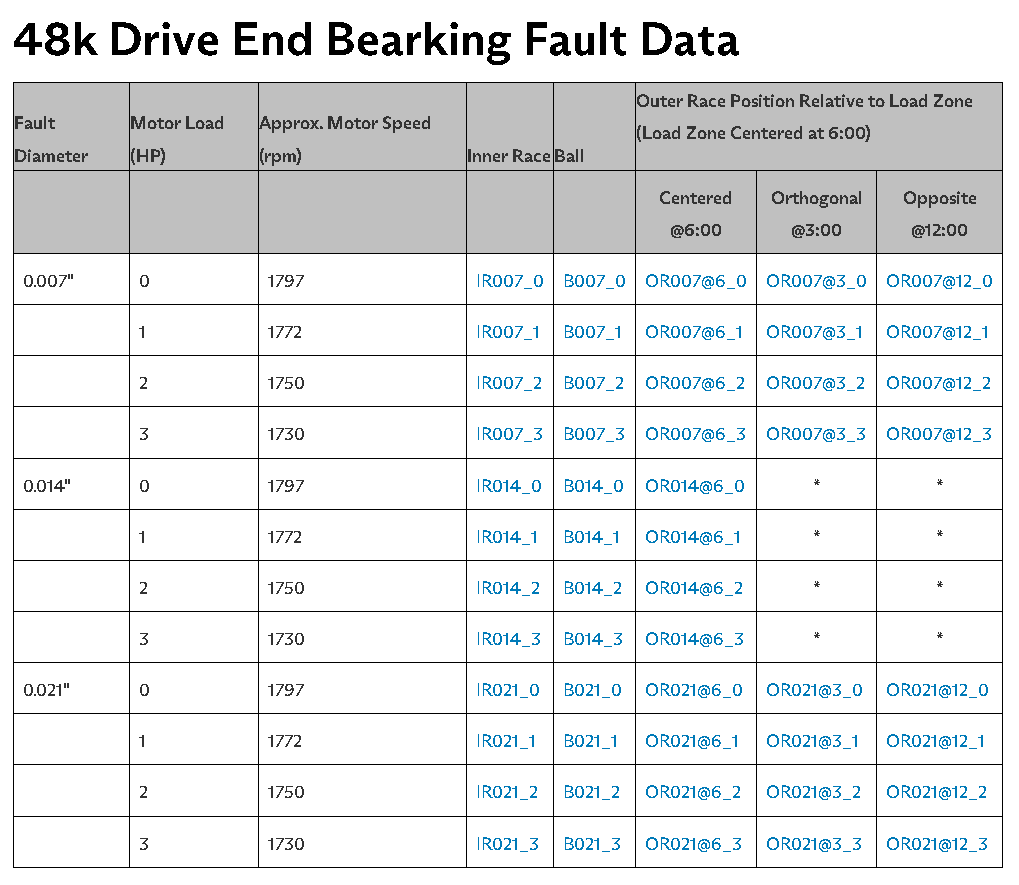
[3. **成熟且丰富的图像处理模型和工具** 14](#_Toc212044161)

1. **数据集详情和数据预处理**
2. **数据集介绍**

本次实验中，数据集采用的是凯斯西储大学的CWRU故障诊断数据集，数据针对正常轴承、单端驱动端和风扇端缺陷进行了收集。驱动端轴承实验的数据采集率为每秒12,000个样本和每秒48,000个样本，采集了约为10s的数据。所有风扇端轴承的数据采集率均为每秒12,000个样本。包含了在不同故障模式和工作条件下的轴承振动信号。该数据集通过电火花加工在轴承上制造故障，并使用加速度传感器采集振动数据。以下为官网数据集展示（不含风扇端缺陷）。



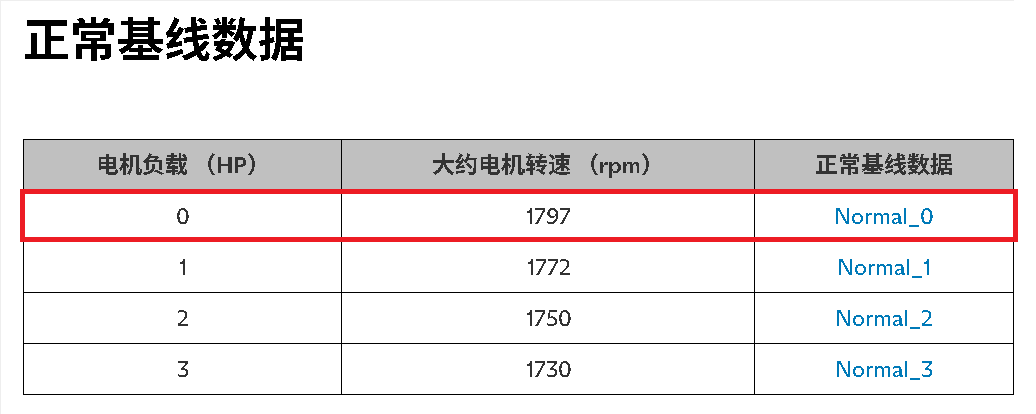




官网数据展示

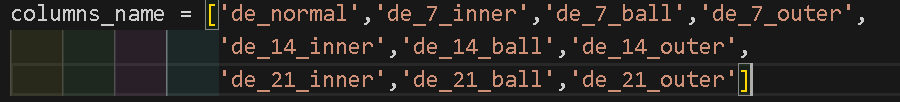
1. 正常数据: 包含没有故障时的基线数据。
2. 故障类型: 数据集主要包括**内圈、外圈**和**滚动体**故障，每种故障都有不同直径的孔（例如7mils、14mils、21mils来模拟。密尔：Mils，一种长度单位）
3. 工况分组: 数据集被组织成多个文件，例如根据采样频率12k、48k和采集位置（Drive End, Fan End）来划分。

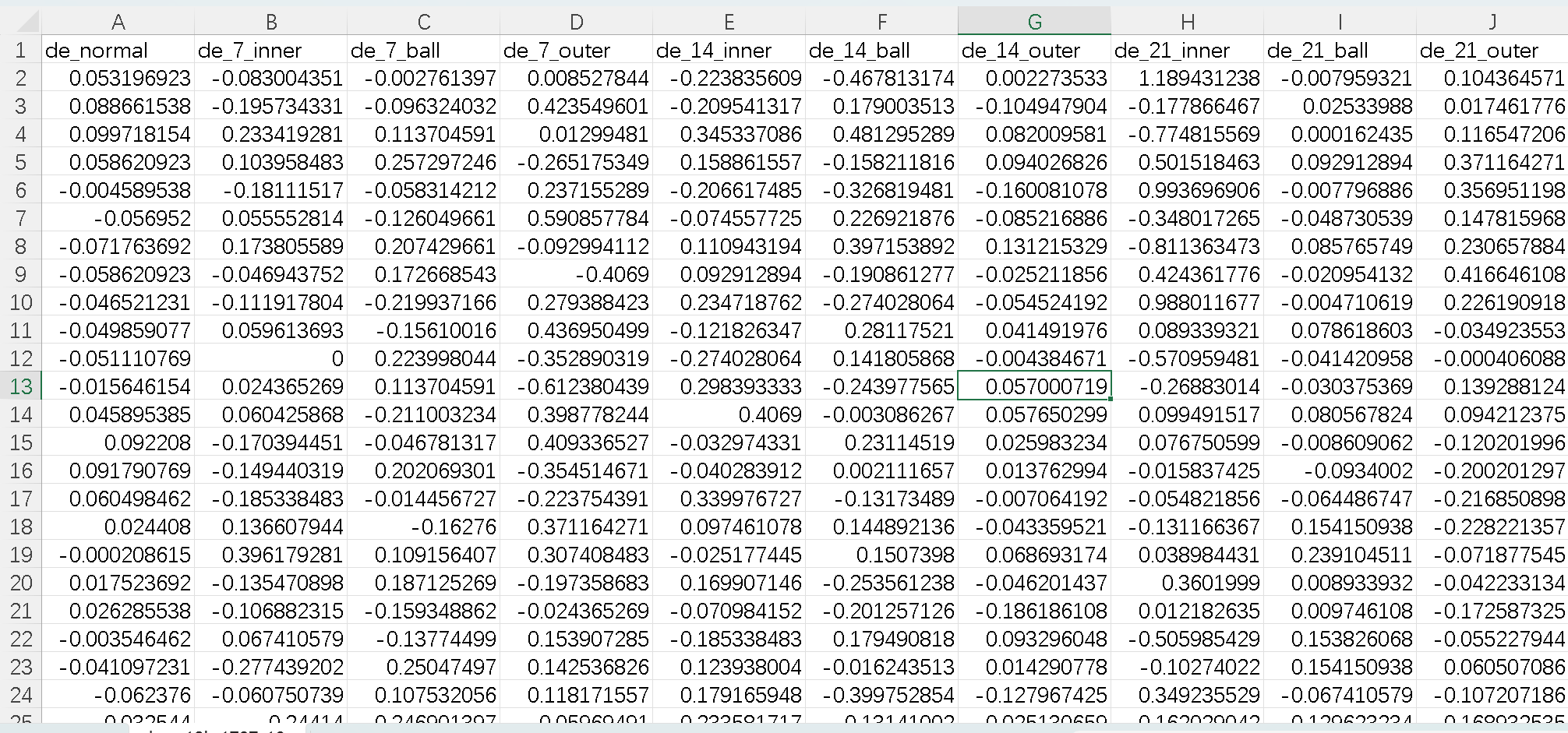
实验采用12K驱动端轴承故障在0 HP条件下的数据,负载外围数据采用居中位置（6：00），分别为十类：1个正常状态和9个不同故障（3种故障类型x 3种故障直径），每一列的数据为121265条, 共有10类数据。



选用数据

数据整理展示，以下为将选取的十类Matlab数据（1个正常状态和9个不同故障）整合为CSV文件，便于后续用于生成图像：

类别名

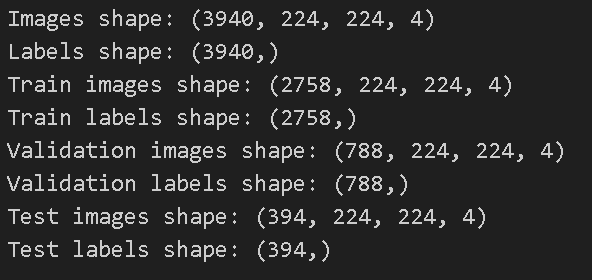


部分数据集

1. **数据集划分**

先使用下列六种算法的任意一种，将一位数据转化为二维图像数据，数据集采用划分比例为 **Train:Val:Test=7:2:1**。

训练集为2758张，验证集为788张，测试集为394张。



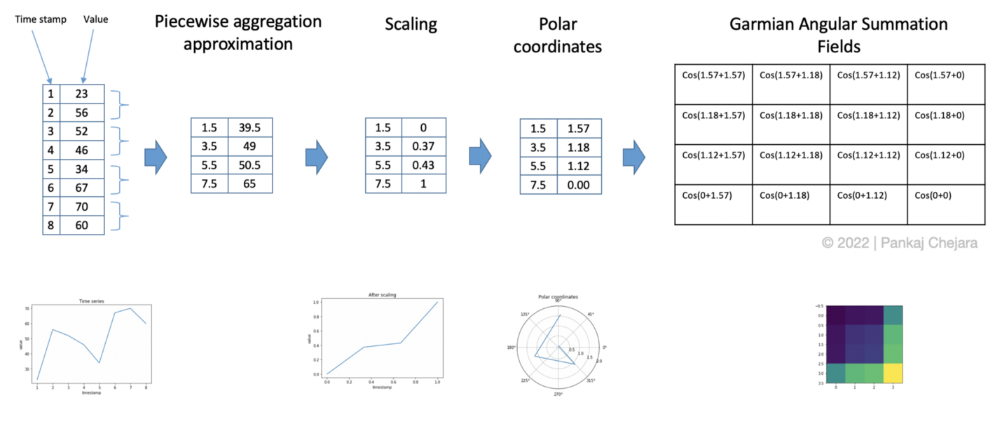
数据集划分后shape

1. **数据预处理算法**

算法方面选用了主流的六种算法。分别为：

* 1. 格拉姆角场（GAF）

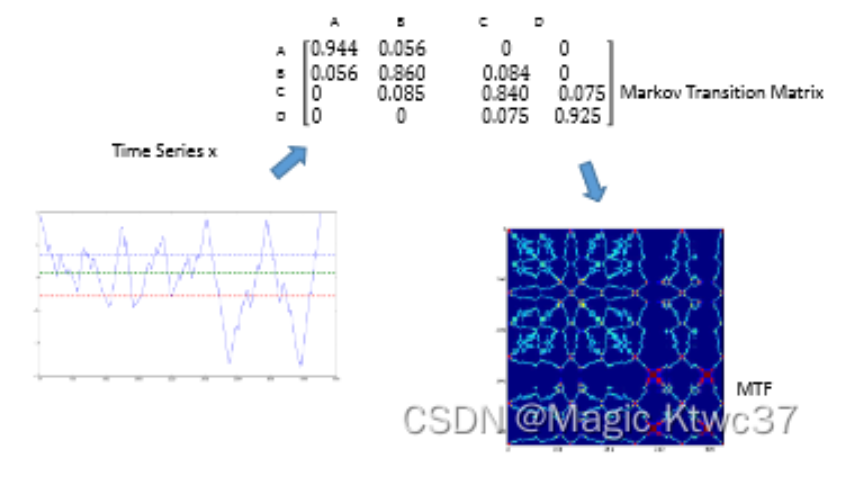
1. 通过取每个 M 点的平均值来聚合时间序列以减小大小。 此步骤使用分段聚合近似 ( Piecewise Aggregation Approximation / PAA)。
2. 区间[0,1]中的缩放值。
3. 通过将时间戳作为半径和缩放值的反余弦（arccosine）来生成极坐标。 这杨可以提供角度的值。
4. 生成GASF / GADF。 在这一步中，将每对值相加（相减），然后取余弦值后进行求和汇总。



格拉姆角场转换流程

* 1. 马尔可夫转换场（MTF）

1. 将时间序列数据离散化为有限数量的状态，并分析这些状态之间的转移关系。
2. 构建一个转移概率矩阵
3. 这些状态的转移概率表示为一个图像。

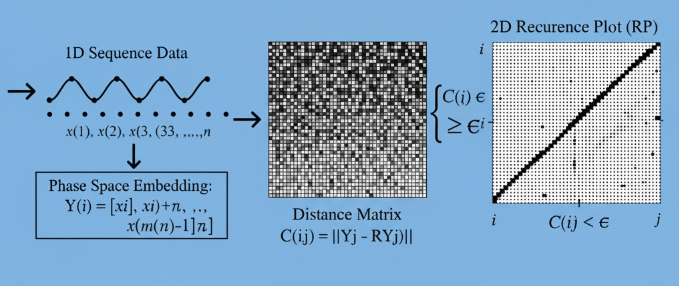


马尔可夫转移场转换流程

* 1. 递归图（RP）

递归图的核心思想基于相空间重构。

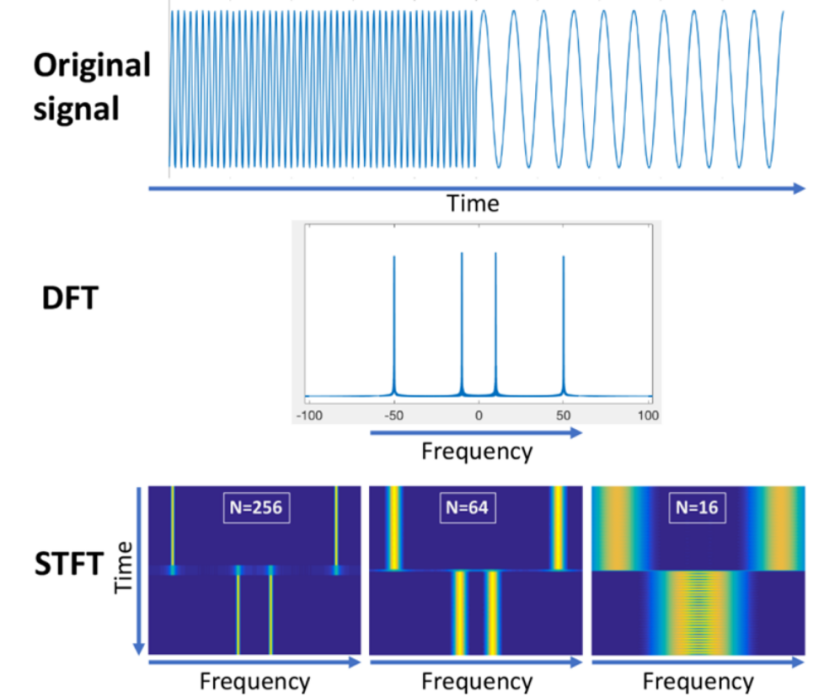
1. 将时间序列数据转换成相空间中的轨迹，然后分析这些轨迹的重现性。
2. 在相空间中，每个点代表系统的一种状态，而轨迹展示了系统状态随时间的变化。
3. 通过观察不同时间点的系统状态是否接近（即轨迹的点是否相邻），可以揭示出系统行为的重复性或周期性。



递归图转换流程

* 1. 短时傅立叶变换（STFT）

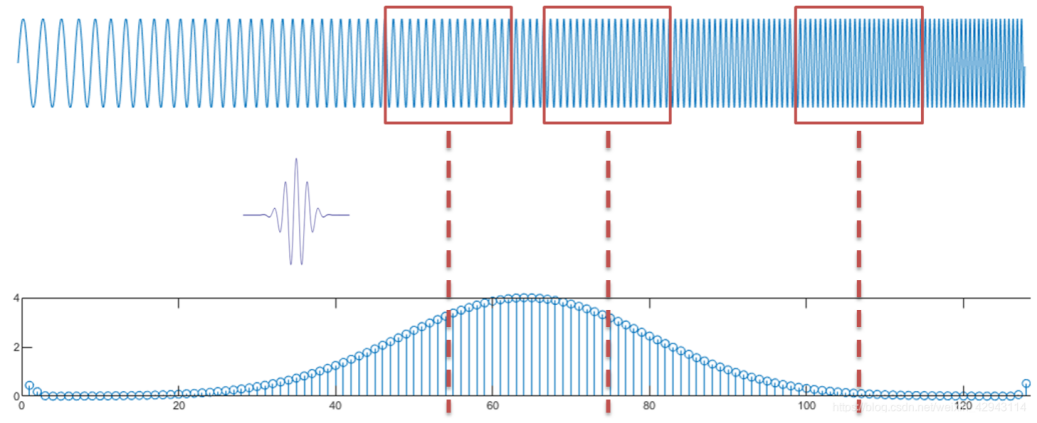
1. 分帧：首先，将待分析的信号分成多个长度相等的窗口，通常使用汉明窗（HammingWindow）来减小边界效应。
2. 加窗：对每个窗口内的数据进行加窗操作，这可以确保时域信号在频域分析时的平滑过渡，减少泄漏效应。
3. 傅立叶变换：对加窗后的每个窗口内的数据进行傅立叶变换，得到频域上的信号表示。
4. 重叠与合并：为了获得更好的时间分辨率，通常将相邻窗口的数据进行一定程度的重叠和合并操作，以平滑信号过渡。



短时傅里叶变换转换流程图（无第四步流程）

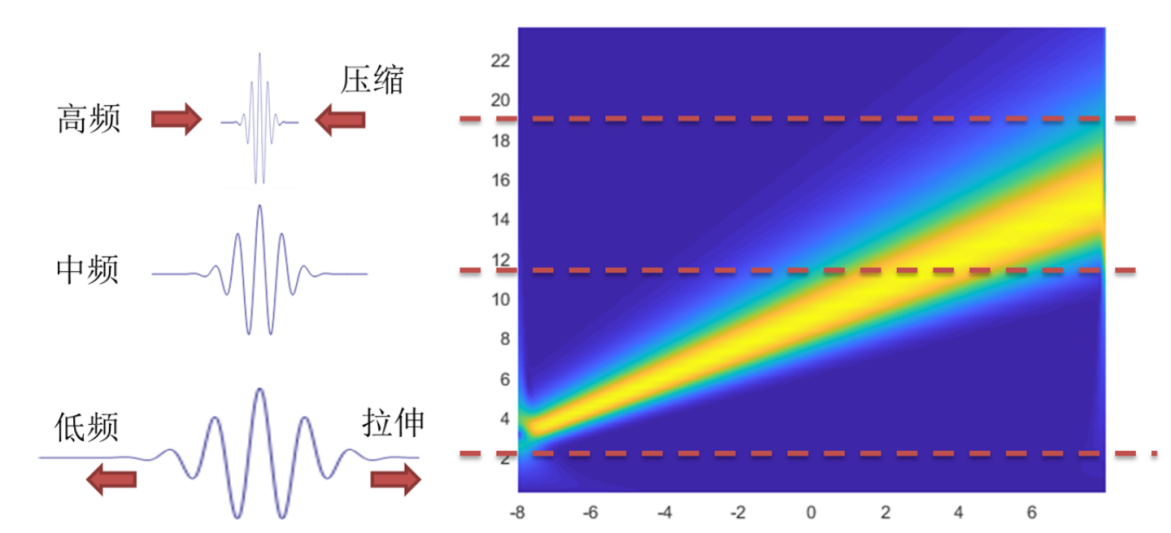
* 1. 连续小波变换(CWT)

1. 在时间域上，可以通过小波在时间上的移动，逐一比较不同位置的窗口信号，得到小波系数，小波系数越大，则证明小波与该段信号的拟合程度越好。计算中用小波函数与该窗口信号的卷积，作为该窗口下的小波系数。窗口的长度和小波的长度是相同的。



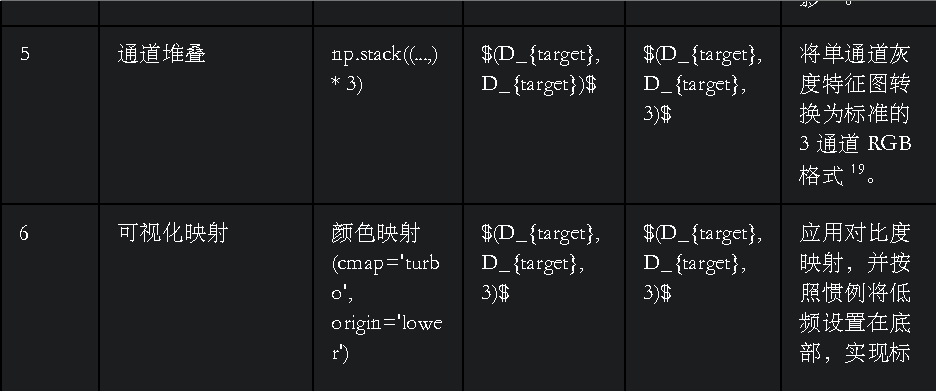
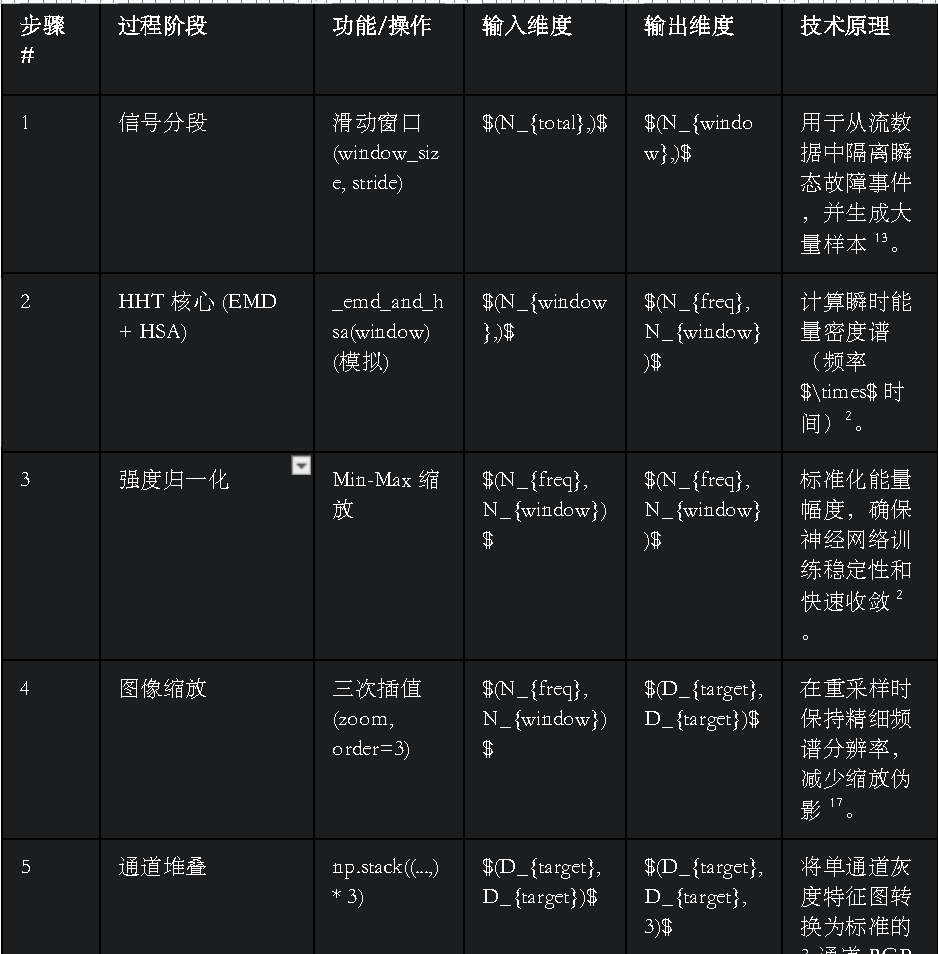
时间域处理流程示意图

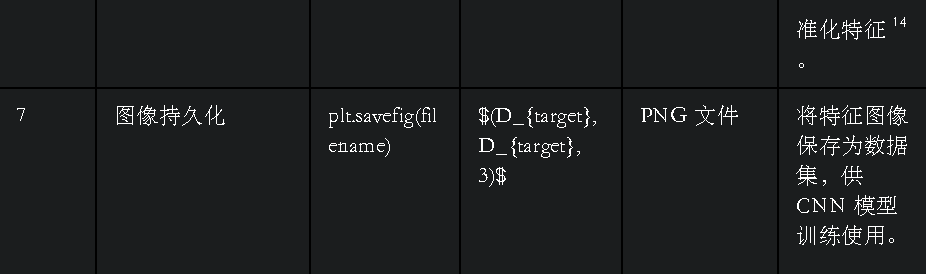
1. 在频率域上，通过拉伸或压缩小波的长度，来改变小波的长短和频率，实现不同频率下的小波系数。相应的，窗口长度也会随着小波长度变化。由于高频处小波被压缩，时间窗变窄，使得时间分辨率更高。
2. 将不同频率下的小波系数组合起来，便得到了时频变换的小波系数图。



频率域处理流程及小波系数组合

* 1. 希尔伯特-黄变换（HHT）





1. **通用图像处理参数**
2. WINDOW\_SIZE决定了生成一张图所需要的数据量，选用大小需要关注RAM的使用情况，2的次方即可，减小窗口大小可以减少RAM使用。（本次实验采用WINDOW\_SIZE=512，则一张图至少需要512\*0.6=307.2条数据，以50ms采集一次数据为例，则需要15.36s可以出一张图。）

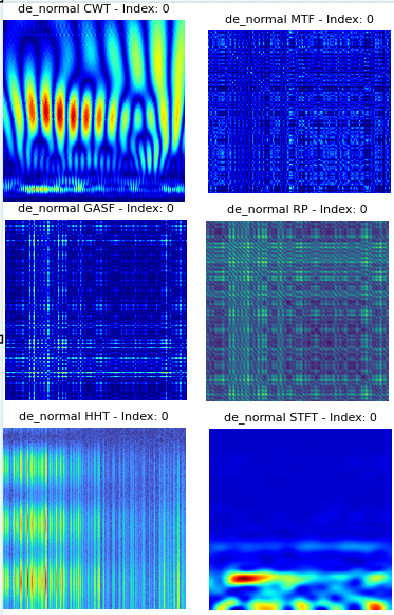
以WINDOW\_SIZE为例，若接口传输来的数据量不足一张图所需要的数据量，则暂且储存起来，等待 **数据量>=WINDOW\_SIZE** 之后，将数据转换为图像。多余的数据量则暂且存储起来，等下一批数据到来之后，数据合流之后再进行判断 **数据量>=WINDOW\_SIZE** 。

1. STRIDE = WINDOW\_SIZE \* (1 - OVERLAP\_RATE)，增加步长可以减少生成的图像数量，其中OVERLAP\_RATE 为生成图像数据重叠的比例。（本次实验统一采用OVERLAP\_RATE=0.4，则步长为307.2。）
2. MAX\_IMAGES\_PER\_COLUM为每列最多生成的图像数量，防止各类数据差异巨大，导致模型在某方面判断异常。本次实验采用值为500张。
3. 图像尺寸，一般情况下，使用算法由一维数据生成的图像的大小即为模型输入要求尺寸最佳。

原理为：若生成的图像太大，在进行模型输入时需要进行压缩，导致部分信息的丢失。若生成图像太小，在进行模型输入时需要拉伸图片，不可避免的导致噪声的出现。例如本次实验采用的是ResNet50模型，模型入口为224\*224，故预处理算法生成图像即为224\*224。

1. **算法生成图像展示**

以下为六种算法生成的图像

****

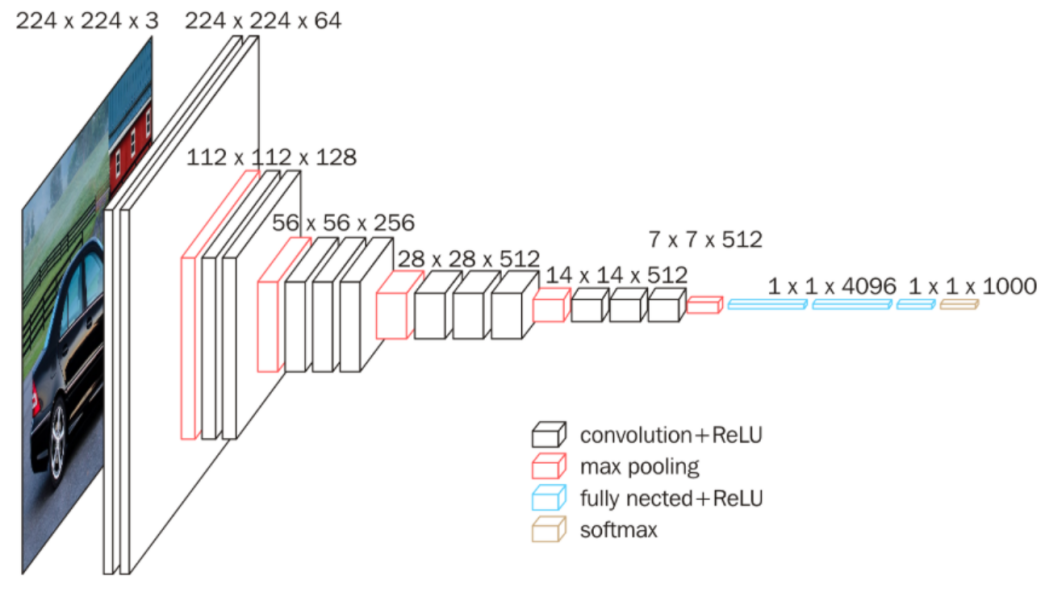
**帧率图展示**

1. **生成图像信息比对**

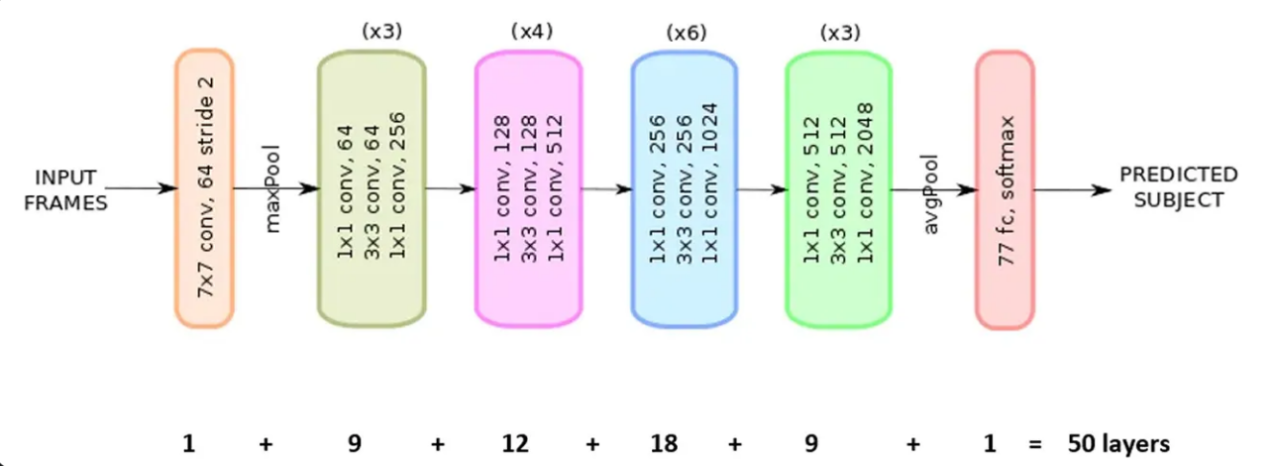
以下为六种转换算法信息比对：

1. 存储占用：存储的占用分别为：**STFT(约23KB) > GAF（约38KB） > CWT(约40KB) >≈MTF(约40KB) > RP(约50KB) > HHT（约59KB），**可得出连续短时傅里叶变换（STFT）的生成图像最小。
2. 生图速度：**RP（约53.5ms）>MTF(约53.6ms)>STFT(约54.6ms)>HHT(约55.6ms)>CWT（约62.8ms）>GAF(约68.2ms)**
3. 图像转化numpy速度：**RP（约7.7ms）>HHT(约7.9ms)>STFT(约8.0ms)>GAF(约8.3ms)>MTF(约8.7ms)>CWT(约12.3ms)**
4. **模型选型及训练策略**

本次实验网络方面选用**VGG16**和**ResNet50**；



VGG16网络结构图

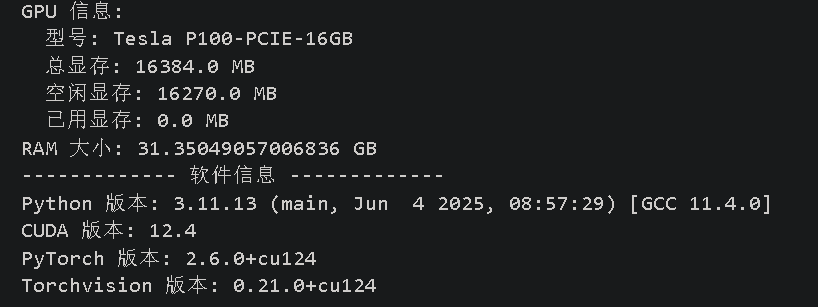


ResNet50网络结构图

1. **训练环境**

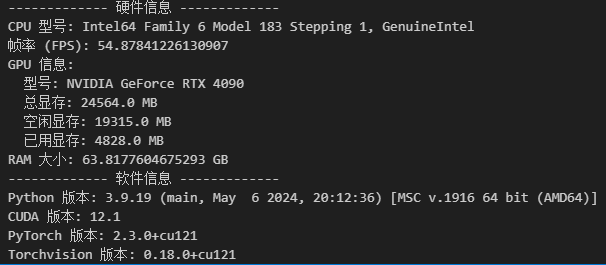
本次实验训练环境分为两种，VGG16（前期基于Kaggle进行实验）与ResNet50两种。

1. Kaggle环境：CPU 型号: x86\_64



Kaggle平台配置

1. 公司深度学习环境：



1. **批次大小**

选择一个可以使RAM占用较高的2的次方数即可。如本次实验中采用batchsize=32。在推理模式中，无论生成的图像是否满足一个批次均可进行推理。

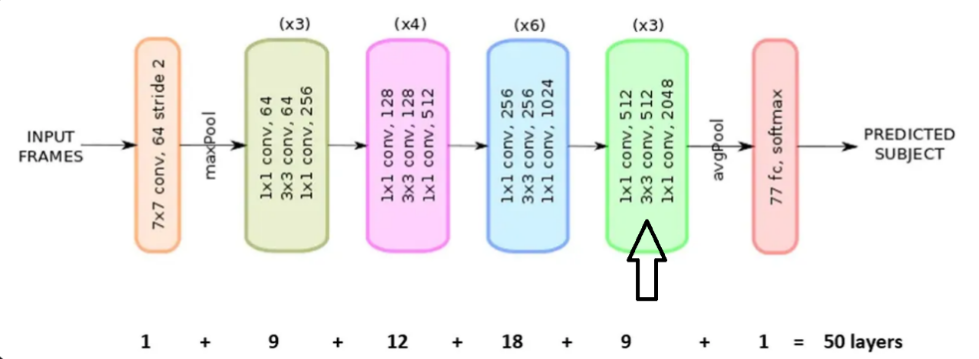
1. **训练轮次（epochs）**

上限200轮较为合适。轮数太低模型可能不能很好的拟合数据特征;轮数太高模型训练时间过长，且容易过拟合。

1. **训练策略**

目前采取以下策略：

1. 学习率调度器策略：对自定义的隐藏层使用较大的学习率（如1e-4,约等于0.0001）；对于已经学习到图像通用特征的预训练模型的解冻层（通用特征如轮廓和纹理等），则采用较小的学习率（如1e-5，约等于0.00001），避免破坏预训练学到的通用特征。
2. 内存清理策略：为防止转换图像时内存超出，采用每50张图像之后，清空一次内存占用。
3. 模型微调策略：通过解冻预训练模型的部分层数来达到更好的训练效果。本次实验中解冻了ResNet50模型的Layer4



解冻层示意图

1. 耐心策略:当模型的验证损失（val\_loss）持续40轮没有减少，则终止训练。
2. 损失函数阈值策略:当模型的验证损失（val\_loss）小于一定值时，终止训练。如常用阈值：0.001。
3. **实验总结**
4. **准确率统计结果**

|  |  |  |  |
| --- | --- | --- | --- |
| **一维转二维深度学习研究** | | | |
| 数据集CWRU | **模型选型** | | |
|  |
| **算法选型** |  | VGG16 | RESNET50 |  |
| STFT | 99.24% | 99.75% |  |
| CWT | 94.59% | 100.00% |  |
| RP | 94.16% | 100.00% |  |
| HHT | 91.88% | 99.49% |  |
| GAFs | 90.20% | 100.00% |  |
| MTF | 86.80% | 100.00% |  |

**注：VGG16模型未进行微调，仅添加了一到两层的隐藏层及Dropout层和最后的分类层。**

1. **模型推理速度**

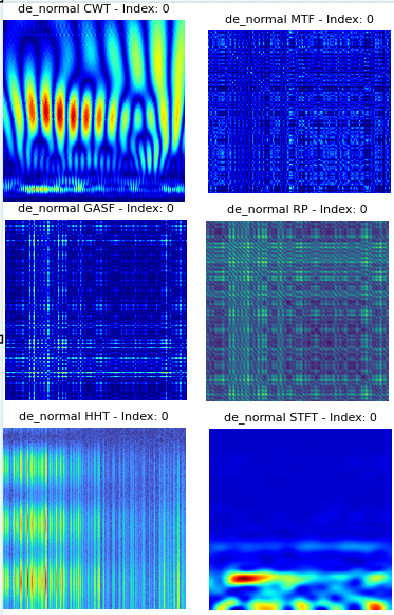
模型的推理速度对比结果如下：GAF(约0.90ms)>STFT(约0.95ms)>RP(约0.95ms)>HHT(约0.98ms)>CWT(约1.20ms)>MTF(约1.28ms)

1. **实验不足点**
2. 仅在一个数据集上进行了实验，且本数据集属于高质量的数据集，实验结果的泛用性不足。如果有较为普通的数据集进行实验，其结果更具泛用性。
3. 开始实验时考虑到实验数据充足，故未采用数据增强操作增加数据量和提高泛化能力。在后续实验中可模拟实验数据不足，通过数据增强，增加噪声，增加数据量，提高模型的泛化能力。

后续需要完善模型评估体系，仅损失函数和准确率不足以反映模型的分类能力。

1. **图像算法生成的图像难以看懂**

虽然二维图像算法令数据可视化，但是高度抽象的算法转化出的图像，大部分人难以区分其NG的原因，具有难以解释性的缺点。如下列部分算法转化的图像所示。

****

1. **一维数据转二维图像进行深度学习的优点**

将一维（1D）数据转换为二维（2D）图像格式进行深度学习，主要是为了能够利用在图像处理领域表现出色的二维卷积神经网络（2D CNN）的强大能力。这种转换带来的主要优势包括：

1. **利用二维卷积神经网络的优势**
2. **特征提取能力：** 2D CNN，尤其是像卷积层（Convolutional Layers）和池化层（Pooling Layers）这样的结构，在处理图像数据时能够高效地捕获局部空间特征、模式和它们之间的关系。
3. **平移不变性：** 2D CNN 通过共享权重（Shared Weights）和池化操作，使其对特征在输入中的位置变化具有一定的容忍度，这有助于模型更好地泛化。
4. **参数共享：** 卷积核在整个输入图像上滑动，重复使用相同的权重，这大大减少了模型的参数数量，降低了计算复杂度，并有助于防止过拟合。
5. **更好地发现隐藏的关联性/结构信息**

一维数据（如时间序列信号）在转换为 2D 图像时，可以通过特定的编码方法.例如，时频分析图如小波图(Wavelet Transform)、递归图(Recurrence Plot)等。将原始信号中不易察觉的复杂特征或多变量之间的相互作用转换为图像中的纹理、形状或颜色变化等空间结构，从而使 2D CNN 能够捕获这些深层次的关联性。

1. **成熟且丰富的图像处理模型和工具**

图像处理和 2D CNN 领域拥有大量成熟、经过充分研究和优化的模型架构（如 AlexNet, VGG, ResNet, Inception 等）以及配套的工具和预训练模型。将 1D 数据转换为 2D 图像可以方便地应用这些现成的强大资源，利用迁移学习来加速训练和提高性能。