****

**《机器学习与振动信号处理》**

**实验报告**

|  |  |
| --- | --- |
| 学 院： | 航海学院 |
| 学　　号： | 2020301020 |
| 姓　　名： | 邱梁城 |
| 专 业： | 信息工程 |
| 实验地点： | 教学东楼D204 |
| 指导教师： | 杨宏晖 |

**西北工业大学**

**2023年6月16日**

目录

[实验六：轴承数据的综合处理 3](#_Toc14737)

[一、 实验目的 3](#_Toc11211)

[二、 实验原理与方法 3](#_Toc5853)

[三、 实验数据说明 4](#_Toc10202)

[四、 实验设计 4](#_Toc30713)

[五、 实验步骤 7](#_Toc20804)

[六、 实验结果 8](#_Toc12189)

[七、 实验讨论 9](#_Toc16473)

# 实验六：轴承数据的综合处理

## 实验目的

1. 能够使用基本的方法对轴承数据进行特征提取，其中包括时域信息，频域信息等等，并能够利用提取的特征搭建神经网络。
2. 构建出自己希望进行处理的课题，并且能够利用课堂所学知识进行分析、处理。并对结果进行分析讨论。

## 实验原理与方法

对信号进行时域特征提取是指在时间域内对信号进行分析，以了解信号在时间上的变化规律。并根据规律提取出一定的特征值，可以从以下方面对信号时域特征提取：

幅值特征：例如最大值、最小值、均值、中位数、峰峰值、均方根值等，可以反映信号的幅值大小和稳定性。

能量特征：例如信号的总能量、不同频率带的能量等，可以反映信号的能量分布特征。

统计特征：例如方差、标准差、偏度、峭度等，可以反映信号的波动性和偏态特征。

分布特征：例如信号的分布参数、分位数等，可以反映信号的分布特征。

时间特征：例如上升时间、下降时间、持续时间等，可以反映信号的时间特征。

这些特征可以从不同角度反映信号的特征，例如信号的幅度、波形、稳定性、偏态和峰态。

与此同时，信号的频率特性也是振动信号中非常关键的一点，他能够在时域信号被噪声淹没的情况下提供有效的信号信息：

频率特征：例如基频、谐波分量、频率带能量等，可以反映信号的频率分布特征。

## 实验数据说明

本次实验采用的是美国凯斯西储大学的轴承公共数据集，西储大学轴承数据是一个用于机器学习和故障诊断研究的公共数据集，其中包含了四种类型的轴承故障（正常、内圈故障、外圈故障、滚动体故障）的振动信号数据。

该数据集包含了来自12个轴承的不同速度和负载下的振动信号数据。每个轴承都有两个加速度传感器，一个安装在水平方向上，一个安装在垂直方向上。每个轴承的振动信号数据都以MATLAB格式存储，并包含了时间序列数据和频域数据。

待检测的轴承支撑着电动机的转轴；  
 驱动端轴承为SKF6205 ，采样频率为12KHz和48KHz；  
 风扇端轴承为SKF6203 ，采样频率为12KHz。

故障设置：

轴承的损伤是用电火花加工的单点损伤；

SKF轴承用来检测直径为0.1778、0.3556、0.5334毫米的损伤；

NTN轴承则是用来检测直径是0.7112、1.016毫米的损伤；

变量名说明：

DE - drive end accelerometer data 驱动端加速度数据

FE - fan end accelerometer data 风扇端加速度数据

BA - base accelerometer data 基座加速度数据（正常）

time - time series data 时间序列数据

RPM- rpm during testing 转每分钟，除以60为旋转频率

## 实验设计

我们根据轴承信号的数据能够以看出，其包含了很多有用的信息，其中能够明显的观测到，他在不同直径下采取了多个数据，包括直径大小为0.007、0.014以及0.021的数据，我们考虑是否可以根据获得到的轴承信号推断出故障直径的大小。

我们首先对轴承信号不同直径大小下的时域进行分析：

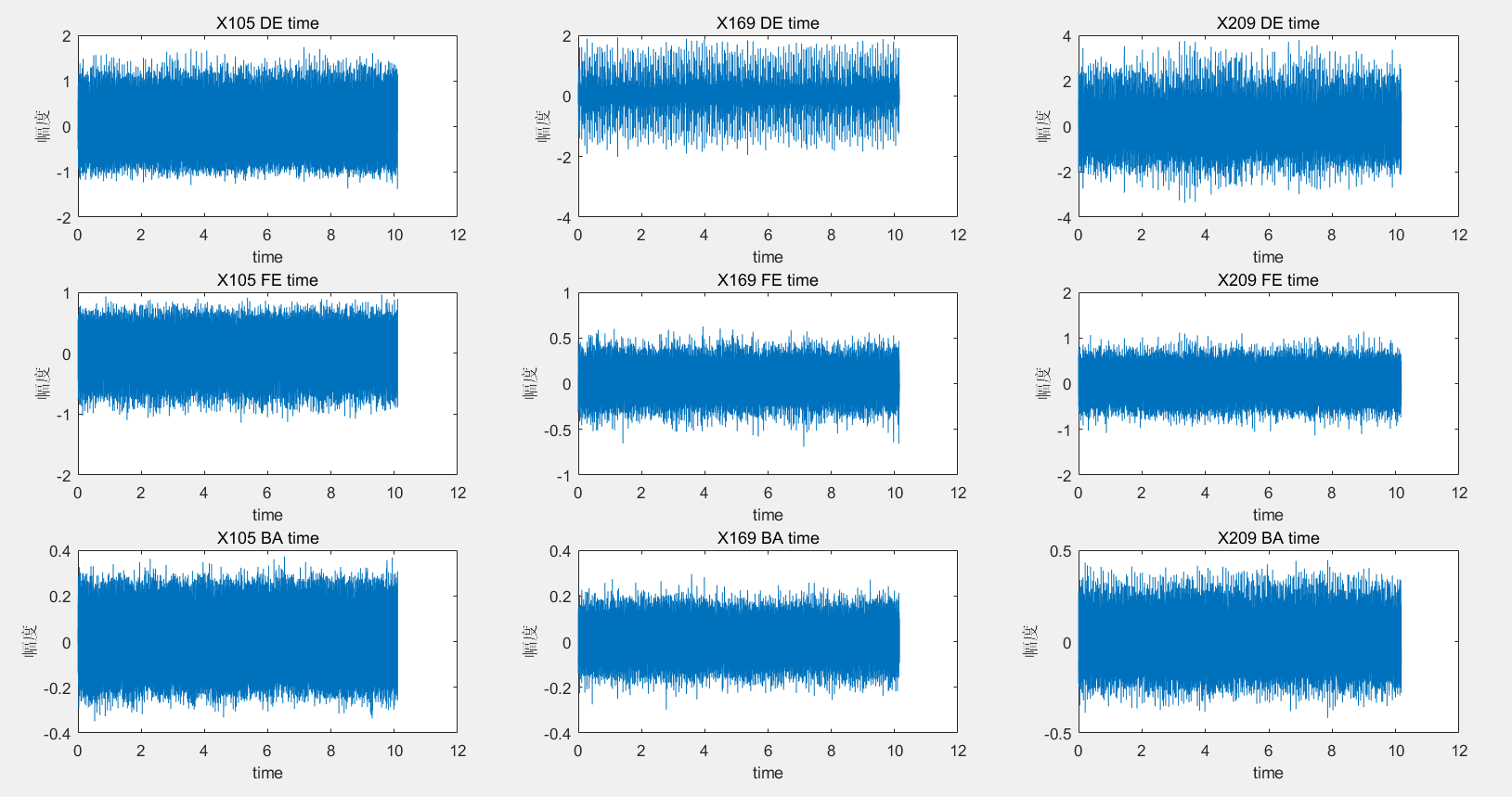


图4.1转速、故障位置不变 故障直径不同 时域图

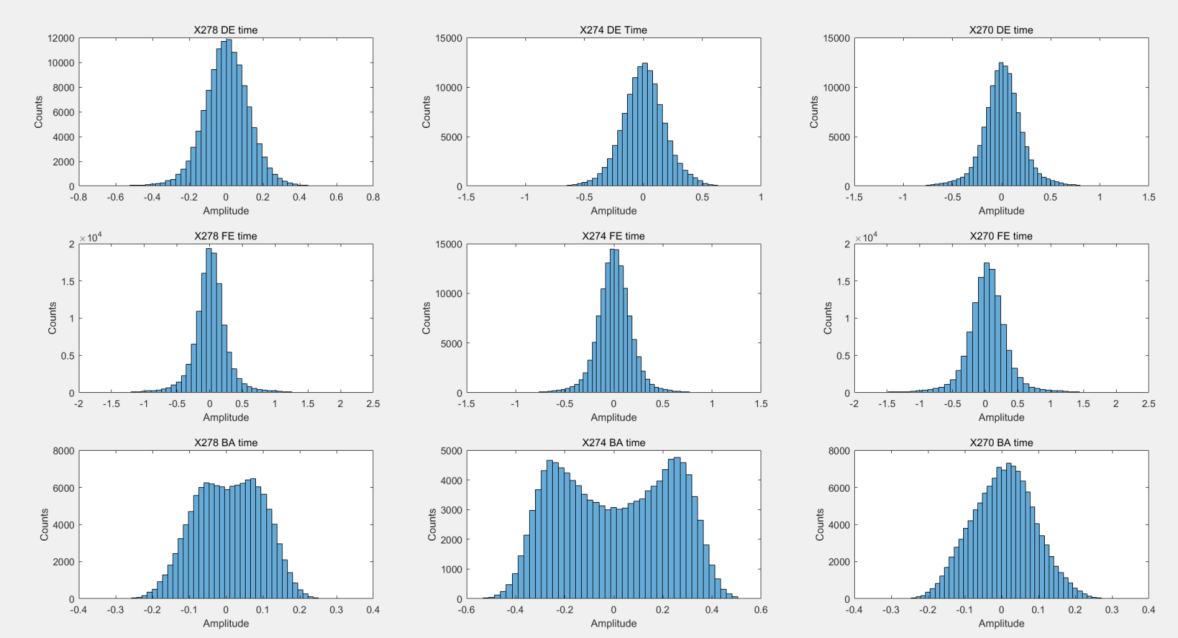


图4.2转速、故障位置不变 故障直径不同 分布图

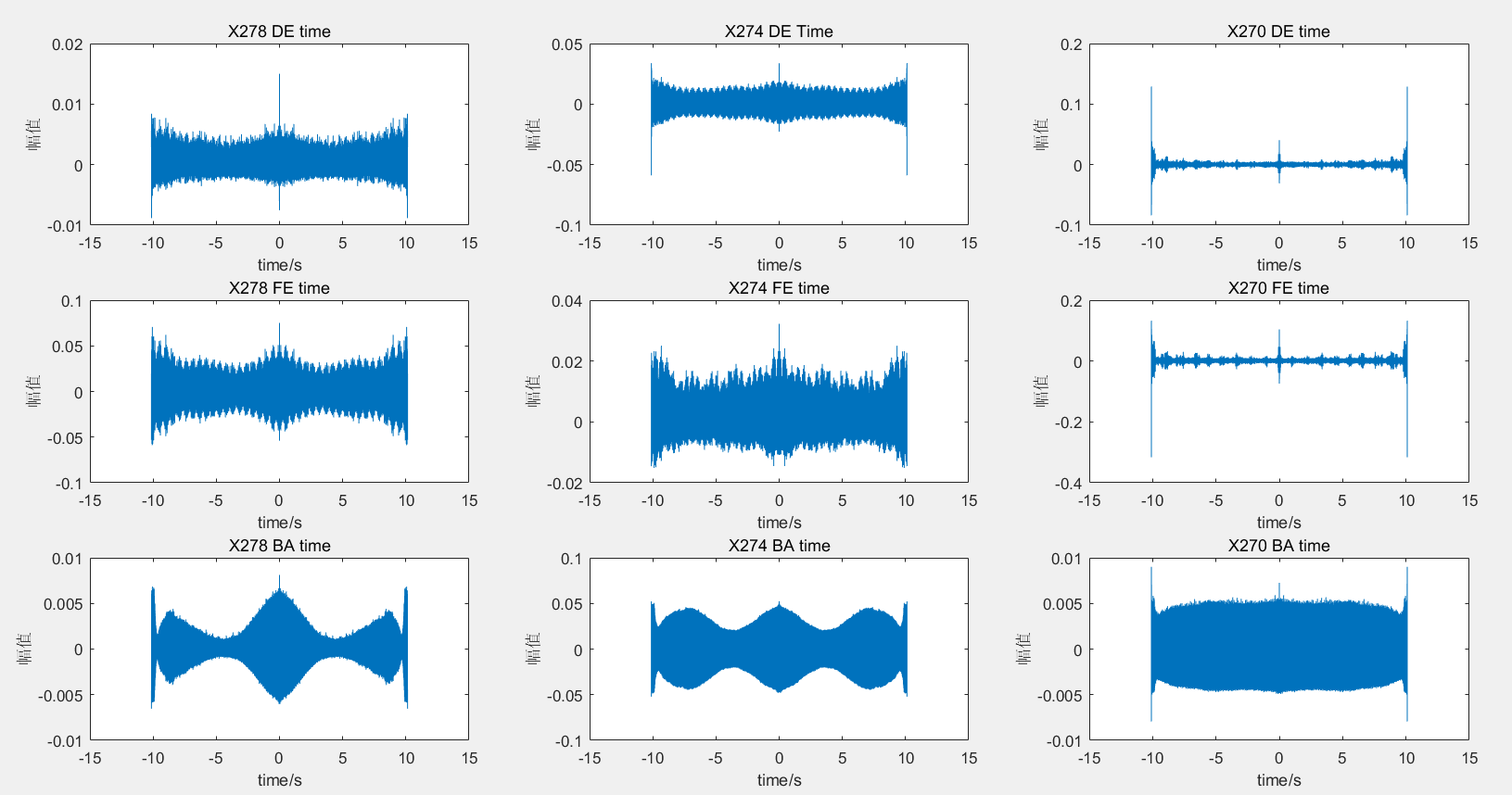


图4.3转速、故障位置不变 故障直径不同 自相关图

可以很明显的看出，，对于不同故障直径大小的轴承信号，其在时域也表现的各不相同，无论从幅值大小、均值、偏锋度以及方差等方面都有较为明显的差异，但是这还远远不够，我们需要挖掘出更具有差异的特征出来，我们考虑频域特性：

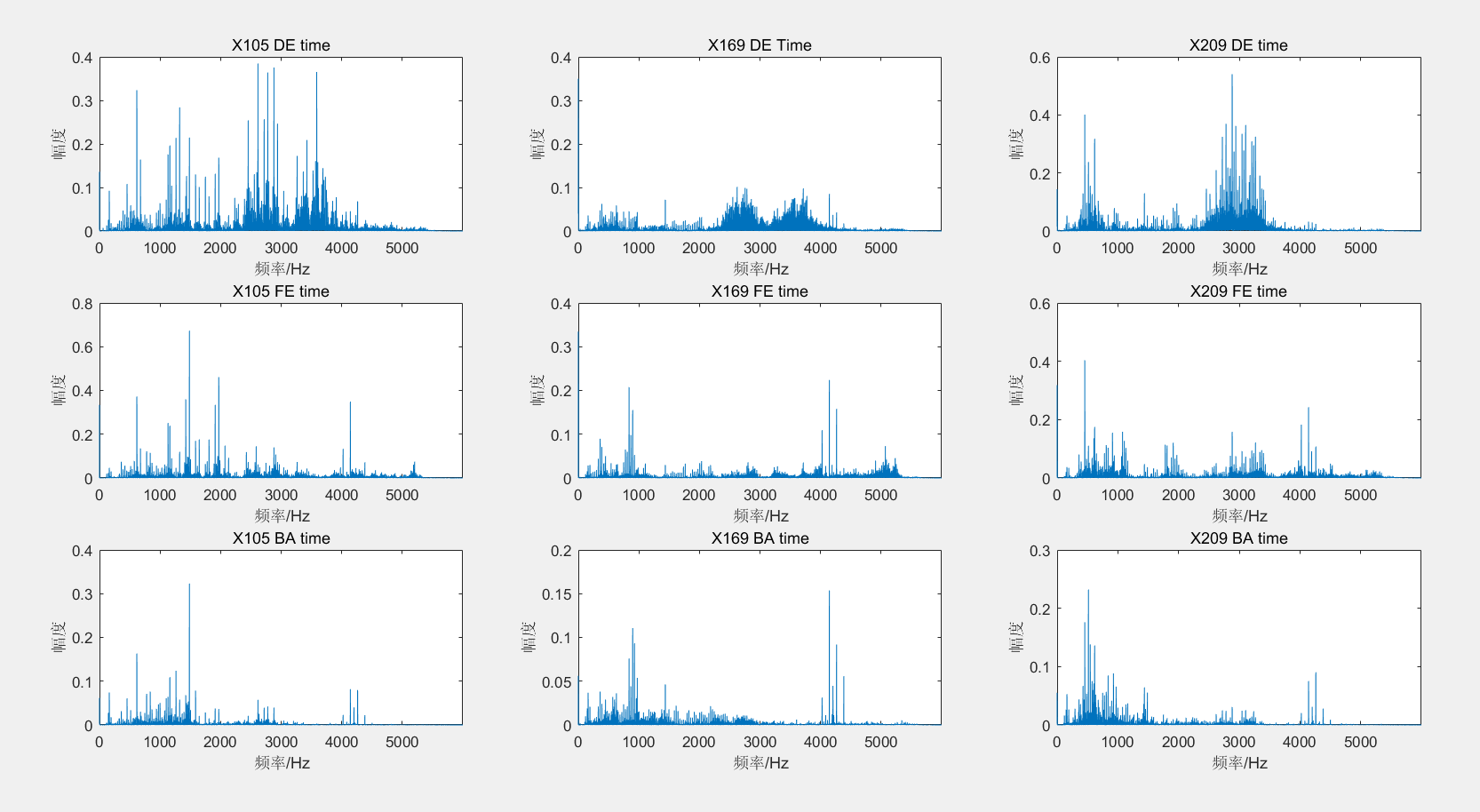


图4.4转速、故障位置不变 故障直径不同 频谱图

能够发现，在不同故障直径下，其频谱差异值非常明显，最高峰的大小、最高峰的频率位置以及频谱平均幅度都是差异较为明显的特征，我们可以依据其进行分类。

至此，我们便有了对希望研究的问题有了一定的理论支撑，于是我们确认所希望研究的问题：通过轴承信号对故障直径大小进行判断！

## 实验步骤

1. 对数据集进行挑选，在这里我们选择12K采样频率下的故障数据中驱动端不同直径大小的数据，总共数据共有52个，分为3类。
2. 数据集的扩充，由于数据集过少，可能会出现过拟合等情况，所以我们对其进行扩充，并进行加噪，使得模型的泛化能力更好，在这里使用滑动窗口，通过矩形窗的移动使得数据集增大。

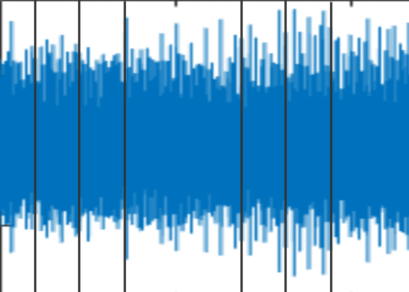


图5-1 滑动窗口对数据进行增强

1. 对于数据集进行特征提取。在这里我们提取19个特征，他们分别是最大值、平均值、最小值、标准差、峰峰值、平均幅值、均方根值、歪度值、峭度值、波形指标、脉冲指标、歪度指标、峰值指标、裕度指标、谱峰幅度值、谱峰频率、谱平均幅度、自相关函数的平均值、自相关函数的标准差。
2. 考虑到这是个简单的三分类问题，我们搭建bp神经网络模型对其进行预测分类，并且我们的输入特征为19，输出3类，根据经验公式确认隐含层节点为：
3. 模型的训练

## 实验结果

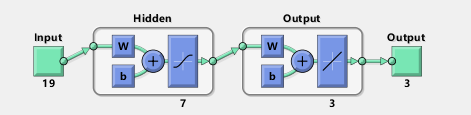


图6-1 神经网络结构

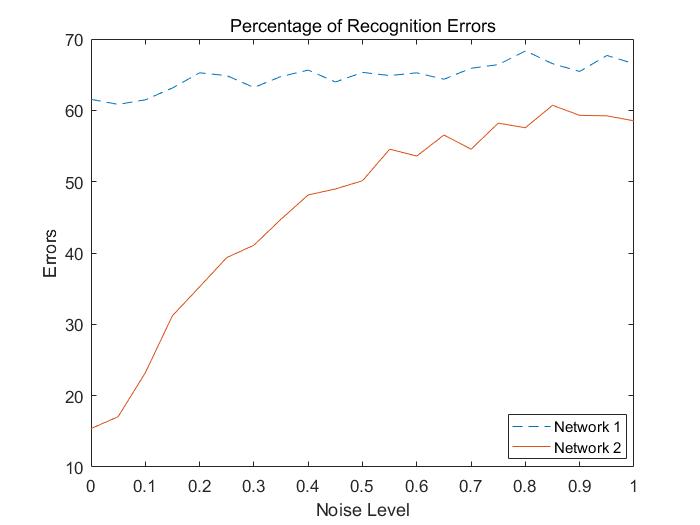


图6-2 原始数据未增强与数据扩充并加噪的训练模型对比

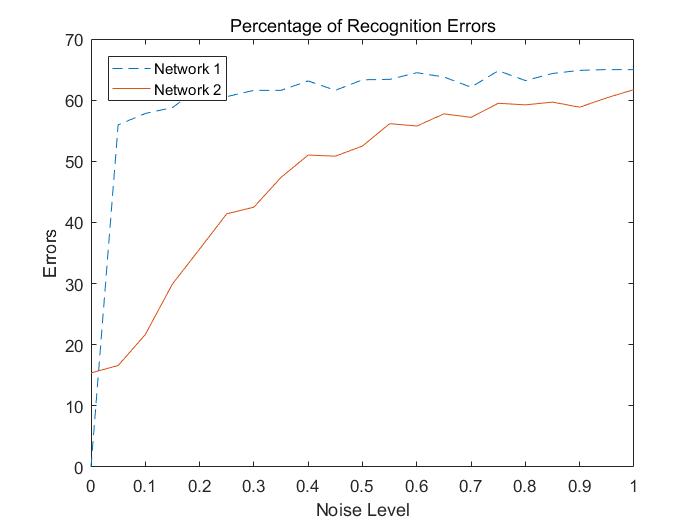


图6-3 对原始数据增强，但没有加噪后的模型准确率

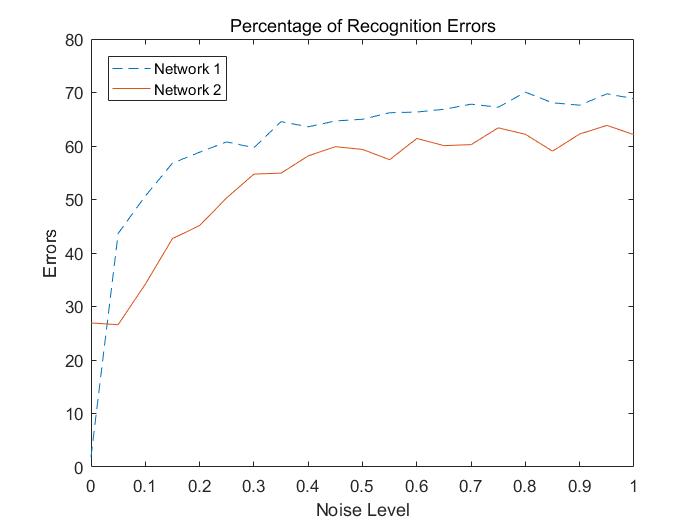


图6-4 只考虑时域特征的模型

## 实验讨论

1. 我们能够发现，在进行训练时，对于数据进行加噪后训练出来的网络具有更好的准确率和泛化性，由于没有进行额外处理的数据训练出的模型其很难对加过扰动后的数据有适应性，所以其准确率并不高。
2. 特征提取的选择和数据集的数量对于这个问题的研究具有十分重要的意义，由于一开始没有考虑频域特征提取，并且只对52个数据进行训练，明显的能够发现其准确率远远低于包含频域特征以及处理后的数据。
3. 由于BP神经网络的结构较为简单，所以其准确率依旧不是特别理想，其原因可能在于我所提取到的特征并没有能够对这个问题有最佳的分类效果，所以我们可以考虑GA优化后的BP神经网络，其效果应该会更佳

附录

clear all; clc

main\_folder\_path = 'G:\matlab\_file\凯斯西储大学数据'; % 替换为实际的文件夹路径

sub\_folders = dir(main\_folder\_path); % 获取主文件夹下的所有子文件夹信息

sub\_folders = sub\_folders(~ismember({sub\_folders.name},{'.','..'})); % 移除.和..文件夹

first\_label=[105,106,107,108,118,119,120,121,130,131,132,133,144,145,146,147,156,158,159,160];

second\_label=[169,170,171,172,185,186,187,188,197,198,199,200];

third\_label=[209,210,211,212,222,223,224,225,234,235,236,237,246,247,248,249,258,259,260,261];

% 遍历所有子文件夹

m = 1;

for k = 1:length(sub\_folders)

fs=12000;

folder\_path = fullfile(main\_folder\_path, sub\_folders(k).name); % 获取子文件夹路径

if(strcmp(folder\_path, 'G:\matlab\_file\凯斯西储大学数据\48k Drive End Bearing Fault Data'))

fs=48000;

end

window\_size = 2\*fs; % 窗口大小为2秒

step\_size = 2\*fs; % 步长为2秒

files = dir(fullfile(folder\_path, '\*.mat')); % 获取所有MAT文件信息

% 遍历所有MAT文件

for i = 1:length(files)

file\_name = files(i).name; % 获取文件名

file\_path = fullfile(folder\_path, file\_name); % 获取文件完整路径

% 读取MAT文件内容

[~, file\_name, ext] = fileparts(file\_path); % 获取文件名和扩展名

num\_str = regexp(file\_name, '\d+', 'match'); % 正则表达式匹配数字部分

num = str2double(num\_str{1}); % 将匹配的数字字符串转换为数字

% 处理文件，例如对data进行操作

data = load(file\_path);

var\_name = ['X', sprintf('%03d', num), '\_DE\_time'];

x\_de\_time = eval(['data.', var\_name]); % 通过eval函数访问动态变量名对应的变量

num\_windows = floor((length(x\_de\_time) - window\_size)/step\_size) + 1; % 计算窗口数

% for j = 1:num\_windows

% start\_index = (j-1)\*step\_size + 1; % 计算窗口起始位置

% end\_index = start\_index + window\_size - 1; % 计算窗口结束位置

% window = x\_de\_time(start\_index:end\_index); % 取出窗口信号

% if(strcmp(folder\_path, 'G:\matlab\_file\凯斯西储大学数据\Normal Baseline Data'))

% T(1,m)=0;

% T(2,m)=1;

% else

% T(1,m)=1;

% T(2,m)=0;

% end

% % 在这里对窗口信号进行处理，例如计算窗口的平均值

% li = feature\_extraction(window, fs);

% x(:, m) = li;

% m = m + 1;

% end

if ismember(num, first\_label)

li = feature\_extraction1(x\_de\_time, fs);

x(:, m) = li;

T(1,m)=1;

T(2,m)=0;

T(3,m)=0;

m=m+1;

elseif ismember(num, second\_label)

li = feature\_extraction1(x\_de\_time, fs);

x(:, m) = li;

T(1,m)=0;

T(2,m)=1;

T(3,m)=0;

m=m+1;

elseif ismember(num, third\_label)

li = feature\_extraction1(x\_de\_time, fs);

x(:, m) = li;

T(1,m)=0;

T(2,m)=0;

T(3,m)=1;

m=m+1;

end

end

clear data

end

X=x;

len=length(X);

% X=x(:,[1:180]);

% T=T(:,[1:180]);

setdemorandstream(pi);

net1 = feedforwardnet(7);

view(net1)

net1.divideFcn = '';

numNoise = 30;

X1 = min(max(repmat(X,1,numNoise),0),1);

T1 = repmat(T,1,numNoise);

net1 = train(net1,X1,T1,nnMATLAB);

numNoise = 30;

Xn = min(max(repmat(X,1,numNoise)+randn(12,len\*numNoise)\*0.1,0),1);

% Xn = repmat(X, 1, numNoise) + randn(19, len\*numNoise) \* 0.2;

Tn = repmat(T,1,numNoise);

net2 = feedforwardnet(7);

net2 = train(net2,Xn,Tn,nnMATLAB);

noiseLevels = 0:.05:1;

numLevels = length(noiseLevels);

percError1 = zeros(1,numLevels);

percError2 = zeros(1,numLevels);

for i = 1:numLevels

Xtest = min(max(repmat(X,1,numNoise)+randn(12,len\*numNoise)\*noiseLevels(i),0),1);

Y1 = net1(Xtest);

percError1(i) = sum(sum(abs(Tn-compet(Y1))))/(len\*numNoise\*2);

Y2 = net2(Xtest);

percError2(i) = sum(sum(abs(Tn-compet(Y2))))/(len\*numNoise\*2);

end

figure

plot(noiseLevels,percError1\*100,'--',noiseLevels,percError2\*100);

title('Percentage of Recognition Errors');

xlabel('Noise Level');

ylabel('Errors');

legend('Network 1','Network 2','Location','NorthWest')