

Whale_Sound

Sound for Anormal Detection

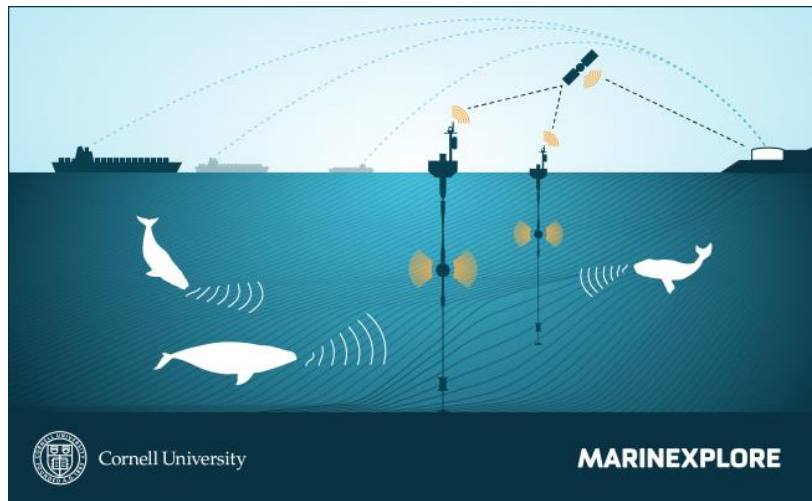
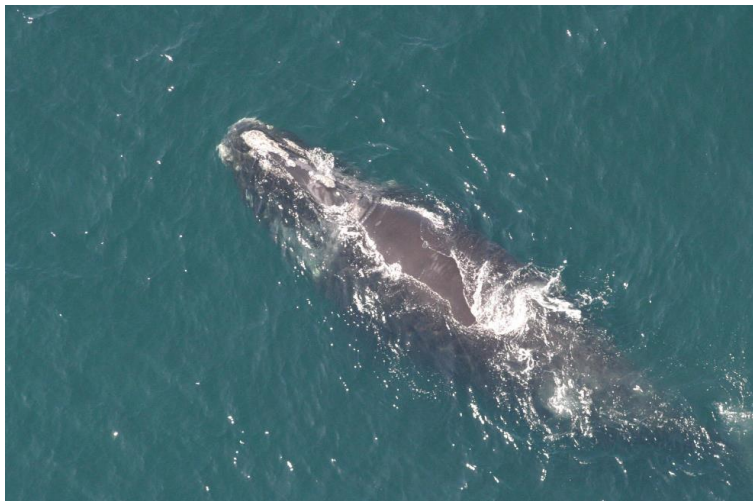
생활의 달인



장비 고장 진단

이 시스템에는 주변소음은 측정되지 않고 모니터링하는 설비의 음만을 추적해 관리하는 사운드 필터법과 시간변화에 대한 주파수 분석이 자동으로 가능하도록 하는 사운드 워터폴 기술이 적용됐다. 하이텍홀딩스 관계자는 “음파를 이용한 발전설비 고장진단장치는 정상상태 기계의 고유한 음과 고장이 진행되는 동안에 발생하는 음의 차이를 주파수로 분석해 설비의 이상 상태를 파악하는 시스템”

Whale Sound Dataset



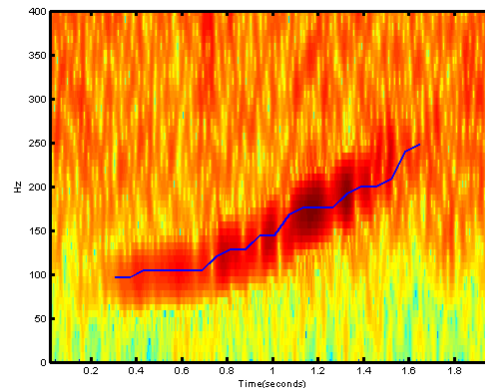
Handing the sound Image

Waveform



FFT

Spectrogram



Python Script for FFT Specgram

```
import aifc

ifn="data/train/train4.aiff"
sf=aifc.open(ifn)
str_frames=sf.readframes(sf.getnframes())
data = np.fromstring(str_frames, np.short).byteswap()
sf.close()

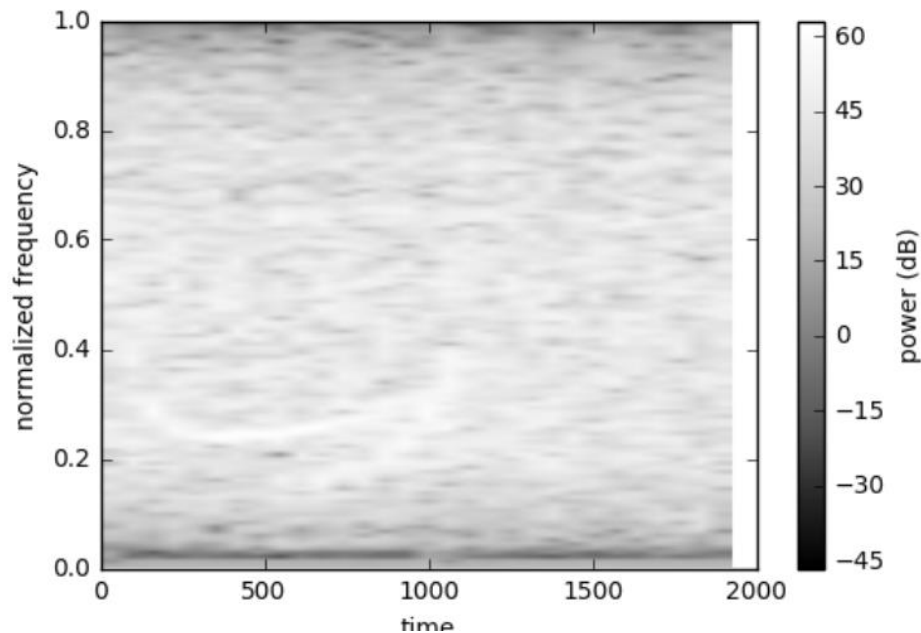
# Use specgram to plot the spectrogram of the data
fig,ax=plt.subplots()
pxx,freq,bins,im=ax.specgram(data,NFFT=256,noverlap=128,cmap='Greys_r')
cb=fig.colorbar(im)
cb.set_label('power (dB)')

plt.xlabel('time')
plt.ylabel('normalized frequency')

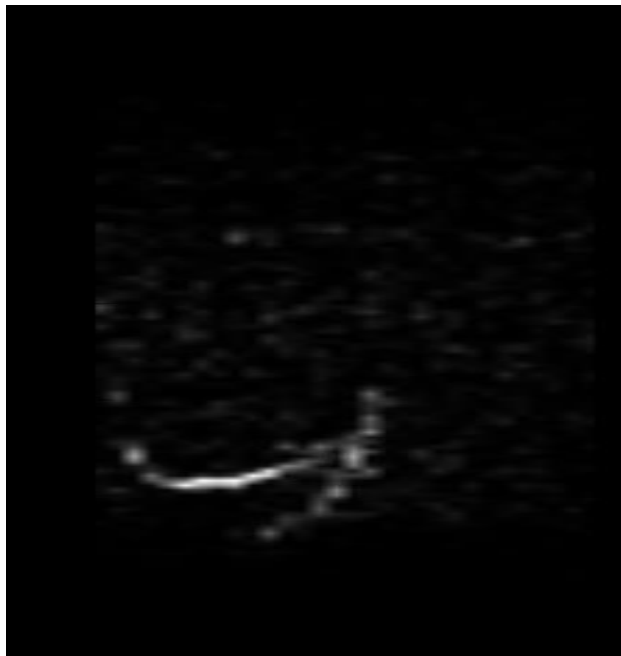
plt.show()
```

FFT for spectrogram

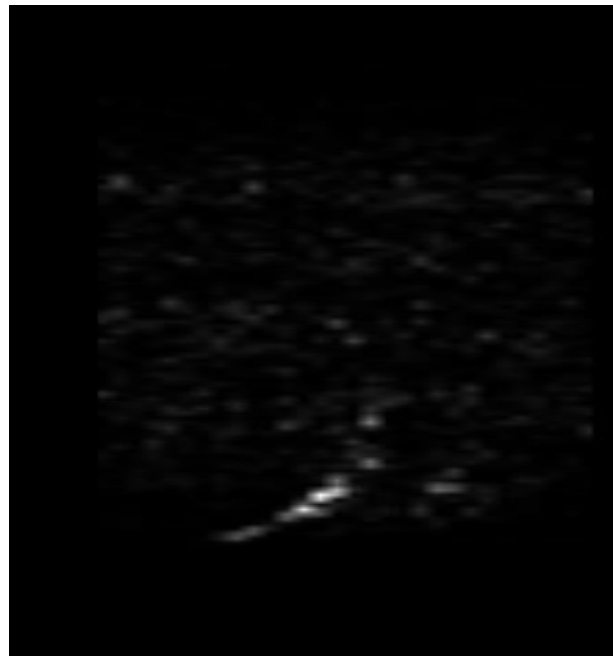
Filename: data/train/train4.aiff
Framerate: 2000
Num Channels: 1
Sample Width: 2
Number of Samples 4000



고래가 아닌 소리



고래 소리



Dataset prepare Method1. labeltext

train.csv in Kaggle

```
clip_name,label
train1.aiff,0
train2.aiff,0
train3.aiff,0
train4.aiff,0
train5.aiff,0
train6.aiff,1
train7.aiff,1
train8.aiff,0
train9.aiff,1
```



- Separate train/val dataset
- Generate label data
- Handle directory
- Handle File extension
- Replace Comma to Space

Label info for caffe

train.txt

```
clip_name,label
data/train/train1.png 0
data/train/train2.png 0
data/train/train3.png 0
data/train/train4.png 0
data/train/train5.png 0
data/train/train6.png 1
data/train/train7.png 1
data/train/train8.png 0
data/train/train9.png 1
```

val.txt

```
clip_name label
data/train/train26863.png 1
data/train/train610.png 0
data/train/train15434.png 0
data/train/train4185.png 0
data/train/train18105.png 0
```

labels.txt

```
Not_whale
whale
```


Bash script

```
%%bash
# Change file extension
sed -i -e 's/aiff/png/g' data/train.csv

# Create training image list
tail -n +2 data/train.csv | sed 's/,/ /g' | awk -v dir=data/train
'{printf("%s/%s %s\n",dir,$1,$2);}' | sed 's/.aiff /.png /g' | head -n 27000 > train.txt

# Create validation image list
tail -n +2 data/train.csv | sed 's/,/ /g' | awk -v dir=data/train
'{printf("%s/%s %s\n",dir,$1,$2);}' | sed 's/.aiff /.png /g' | tail -n -3000 > validate.txt

# Create labels file
rm -f labels.txt
echo "not-whale" >> labels.txt
echo "whale" >> labels.txt
```

Dataset prepare Method2. subfolder

DIGITS

train.csv

/Dataset

train1.png
train1.png
train2.png
train3.png
train4.png
train5.png
train6.png
train7.png
train8.png
train9.png



- Make subfolder for each class
- Move each image files

/Dataset

/not_whale

train1.png
train1.png
train2.png
train3.png
train4.png
train5.png
train8.png

/whale

train6.png
train7.png
train9.png

Python script for moving files

```
import os
import pandas as pd

# filelist
train = pd.read_csv('train-png.csv', index_col='clip_name')
#foldername of class
whaleIDs = list(train['label'].unique())

#make subdirectory for class
os.makedirs('./train-sub/'+w)

# copy image from original folder to sub folder
for image in train.index:
    folder = train.loc[image, 'label']
    old = './train-old/{0}'.format(image)
    new = './train-sub/{0}/{1}'.format(folder, image)
    try:
        os.rename(old, new)
    except:
        print('{0} - {1}'.format(image, folder))
```

Prepare LMDB

DIGITS

/Dataset

/not_whale

train1.png
train1.png
train2.png
train3.png
train4.png
train5.png
train8.png

/whale

train6.png
train7.png
train9.png

The screenshot shows the 'New Image Classification Dataset' form in the DIGITS interface. The form is divided into several sections:

- Image Type:** A dropdown menu set to 'Grayscale'.
- Image size:** Two input fields, both set to '256'.
- Resize Transformation:** A dropdown menu set to 'Squash'.
- Buttons:** A 'See example' button.
- Use Image Folder / Use Text Files:** Two tabs, with 'Use Image Folder' selected.
- Set:** A section with a checkbox 'Use local paths on server' which is checked.
- Training:** A text input field containing '/home/ubuntu/data/whale/train.txt'.
- Image folder (optional):** A text input field containing '/home/ubuntu/data/whale/'.
- Validation:** A checkbox which is checked, with a text input field containing '/home/ubuntu/data/whale/validate.txt'.
- Test:** A checkbox which is unchecked, with a text input field containing 'enter path'.
- Shuffle lines:** A dropdown menu set to 'Yes'.
- Labels:** A text input field containing '/home/ubuntu/data/whale/labels.txt'.
- DB backend:** A dropdown menu set to 'LMDB'.
- Image Encoding:** A dropdown menu set to 'PNG (lossless)'.
- Dataset Name:** A text input field containing 'whale_sound'.
- Buttons:** A 'Create' button.

LMDB

Training

Alexnet

Googlenet

DIGITS

New Model

Version 3.0.0-rc.3

New Image Classification Model

Select Dataset ⓘ

whale_sounds

☐ Use client side file

Python Layer File (server side) ⓘ

whale_sounds

Done 01:59:00 PM

Image Size

256x256

Image Type

GRAYSCALE

DB backend

Imdb

Create DB (train)

27000 images

Create DB (val)

3000 images

Data Transformations

Crop Size ⓘ

none

Subtract Mean ⓘ

Image

Solver Options

Training epochs ⓘ

5

Snapshot interval (in epochs) ⓘ

1

Validation interval (in epochs) ⓘ

1

Random seed ⓘ

[none]

Batch size ⓘ

[network defaults]

Solver type ⓘ

Stochastic gradient descent (SGD)

Base Learning Rate ⓘ

0.01

☐ Show advanced learning rate options

Standard Networks

Previous Networks

Custom Network

Caffe

Torch (experimental)

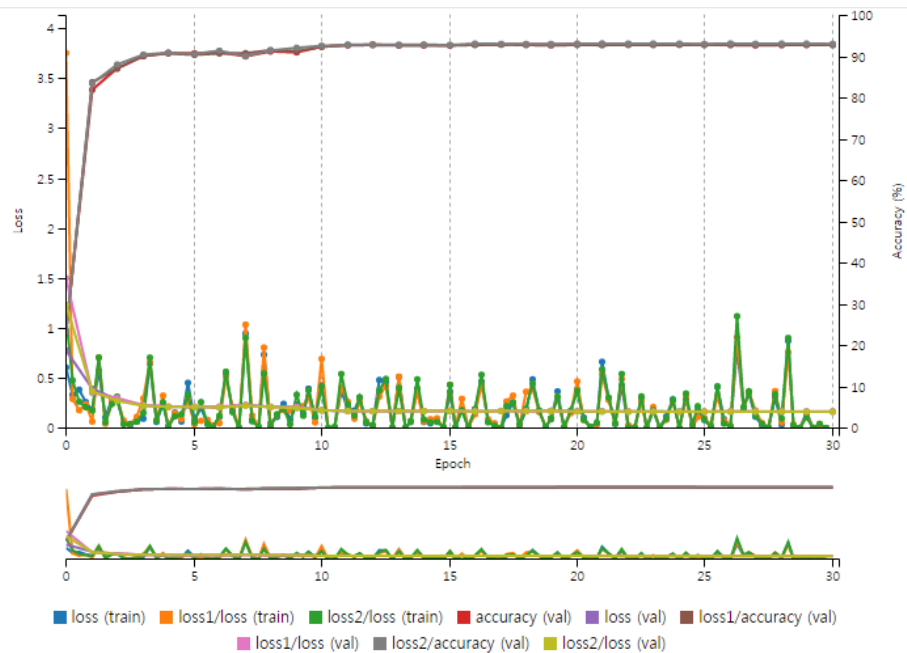
Network	Details	Intended image size
<input type="radio"/> LeNet	Original paper [1998]	28x28 (gray)
<input checked="" type="radio"/> AlexNet	Original paper [2012]	256x256 Customize
<input type="radio"/> GoogLeNet	Original paper [2014]	256x256

Model Name ⓘ

whale_sounds_baseline

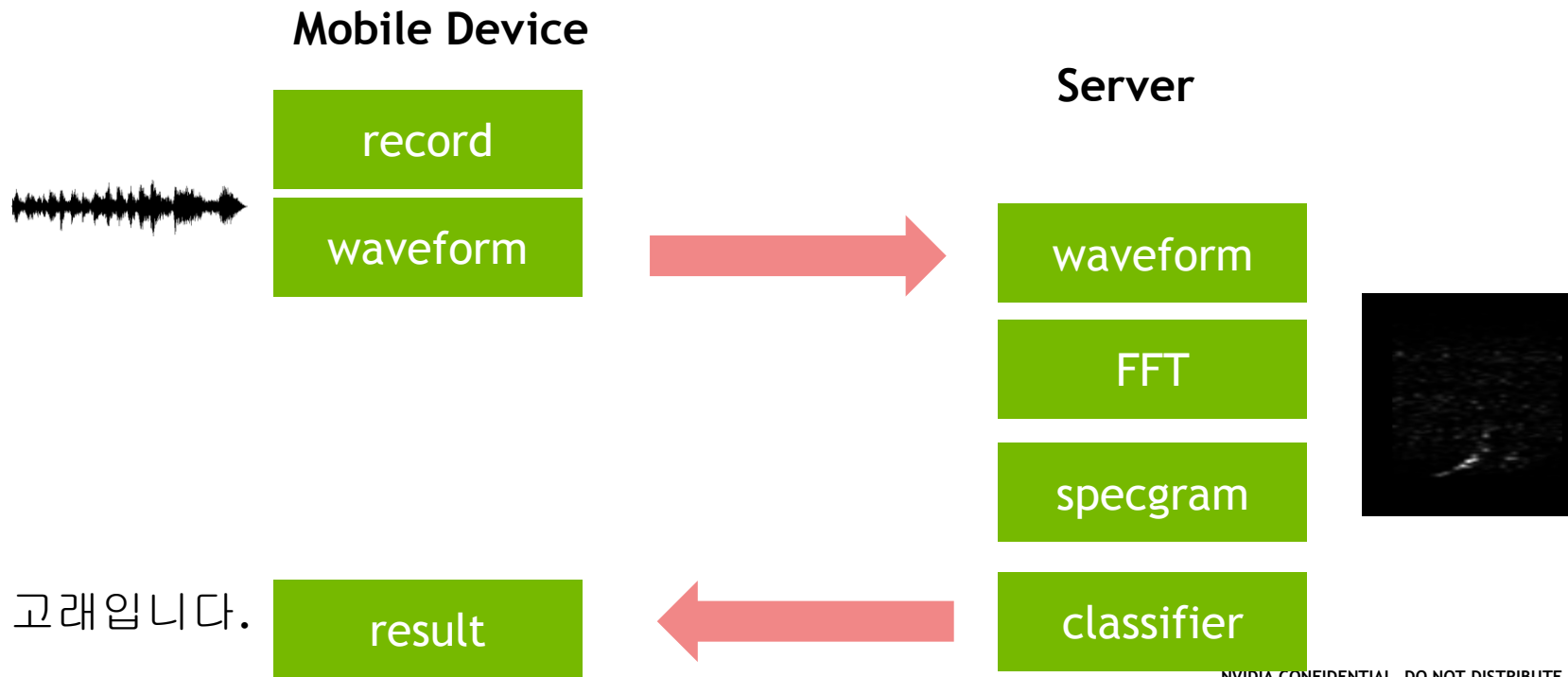
Create

result



30	
accuracy (val)	93.0037
loss (val)	0.165142
loss1/accuracy (val)	92.8172
loss1/loss (val)	0.16607
loss2/accuracy (val)	93.1103
loss2/loss (val)	0.164063

Deployment



Classifier in python

```
import numpy as np
import matplotlib.pyplot as plt
import caffe

MODEL_JOB_NUM = '20170120-092148-8c17'
DATASET_JOB_NUM = '20170120-090913-a43d'

MODEL_FILE = '/home/ubuntu/digits/digits/jobs/' + MODEL_JOB_NUM + '/deploy.prototxt'
PRETRAINED = '/home/ubuntu/digits/digits/jobs/' + MODEL_JOB_NUM + '/snapshot_iter_32270.caffemodel'
MEAN_IMAGE = '/home/ubuntu/digits/digits/jobs/' + DATASET_JOB_NUM + '/mean.jpg'

net = caffe.Classifier(MODEL_FILE, PRETRAINED, channel_swap=(2,1,0), raw_scale=255, image_dims=(256, 256))

prediction = net.predict([inputimg])
```


aiff2FFT in python

```
def aiff2amplitudes(aiff_path):
    s = aifc.open(aiff_path, 'r')
    nframes = s.getnframes() #The total number of audio frames in the file
    strsig = s.readframes(nframes)
    return np.fromstring(strsig, np.short).byteswap()

def amplitudes2spectrogram(amplitudes):
    return_data = plt.specgram(amplitudes,NFFT=256,noverlap=128)
    pxx = return_data[0]
    return pxx

def convert(image_folder):
    for root, dirs, filenames in os.walk(image_folder):
        for f in filenames:
            if os.path.splitext(f)[-1] == '.aiff':
                amplitudes = aiff2amplitudes(os.path.join(root,f))
                spectrogram = amplitudes2spectrogram(amplitudes)
                spectrogram=(spectrogram-np.min(spectrogram))/np.max(spectrogram)
                out_name = os.path.splitext(f)[-2] + '.png'
                io.imsave(os.path.join(root,out_name),spectrogram)
                print out_name
```