建立 Kaldi gstream Online server

從 GitHub 上下載必要文件

```
gst-kaldi-nnet2-online

kaldi-gstreamer-server

kaldi
```

編譯 kaldi 程式，編譯方式如下

```
speech@GPU:~/Desktop$ git clone https://github.com/kaldi-asr/kaldi.git

speech@GPU:~/Desktop$ cd kaldi/tools

speech@GPU:~/Desktop/kaldi/tools$ extras/check_dependencies.sh

speech@GPU:~/Desktop/kaldi/tools$ make -j 12

speech@GPU:~/Desktop/kaldi/tools$ extras/install_irstlm.sh

speech@GPU:~/Desktop/kaldi/tools$ cd ../src/

speech@GPU:~/Desktop/kaldi/src$ ./configure --shared

speech@GPU:~/Desktop/kaldi/src$ make depend -j 12

speech@GPU:~/Desktop/kaldi/src$ make –j 12

speech@GPU:~/Desktop/kaldi/src$ make ext
```

編譯 kaldi-plugin 程式，編譯方式如下

```
speech@GPU:~/Desktop/kaldi/src/gst-plugin$ sed -i 's/-lmkl_p4n//g' Makefile

speech@GPU:~/Desktop/kaldi/src$ make depend -j 12

speech@GPU:~/Desktop/kaldi/src$ make -j 12
```

參考網址如下

https://github.com/alumae/kaldi-gstreamer-server
https://github.com/alumae/gst-kaldi-nnet2-online
https://github.com/kaldi-asr/kaldi
http://jrmeyer.github.io/asr/2016/01/26/Installing-Kaldi.html

啟動辨認器

如以下步驟已經建立完，請直接執行 start.sh，執行方式如下：

```
sudo bash start.sh -y kaldi_models/formosa.yaml
```

如需要停止，請執行：

```
sudo bash stop.sh
```

如果是第一次建立系統，啟動前請先建立和修改幾個部分：辨認器和運行執行檔

建立運行執行檔(start.sh)

請注意 PORT、CERTFILE 和 KEYFILE 變數(黃色標記部分)。port 號請不要有衝突，另外由於學校新制需使用安全連線(SSL)的方式才可以進行校外連線，因此我們將連線所需的加密檔案連接至我們的辨認器，我們使用 apache 裡已經有的預設檔。

再來還有路徑的部分(綠色標記部分)，請修改成自己的位置。

```
nano start.sh

--------------------------------------------------------------------------------

#!/bin/bash



MASTER="localhost"

PORT=8085

CERTFILE='/etc/apache2/ssl/apache.crt'

KEYFILE='/etc/apache2/ssl/apache.key'



usage(){

  echo "Creates a worker and connects it to a master.";

  echo "If the master address is not given, a master will be created at localhost:80";
```

```
  echo "Usage: $0 -y yaml_file [-m master address] [-p port number]";
}

while getopts "h?m:p:y:" opt; do
    case "$opt" in
    h|\?)
        usage
        exit 0
        ;;
    m)  MASTER=$OPTARG
        ;;
    p)  PORT=$OPTARG
        ;;
    y)  YAML=$OPTARG
        ;;
    esac
done

#yaml file must be specified
if [ "$YAML" == "" ] ; then
  usage;
  exit 1;
fi;

if [ "$MASTER" == "localhost" ] ; then
  # start a local master
  python /home/brian/ASR-sysytem/kaldi-gstreamer-server/kaldigstserver/master_server.py --port
=$PORT --certfile=$CERTFILE --keyfile=$KEYFILE 2> /home/brian/ASR-sysytem/master_ner.log &
fi
```

```
#start worker and connect it to the master

export GST_PLUGIN_PATH=/home/brian/ASR-sysytem/gst-kaldi-nnet2-online/src #/:/home/brian/ASR-sysytem/kaldi/src/gst-plugin/


python /home/brian/ASR-sysytem/kaldi-gstreamer-server/kaldigstserver/worker.py -c $YAML -u wss://$MASTER:$PORT/worker/ws/speech 2> /home/brian/ASR-sysytem/worker_ner.log &
```

建立運行執行檔(stop.sh)

這邊不需要做任何修改，它會將**所有執行**的辨認器關掉(請小心使用)

```
nano stop.sh

-------------------------------------------------------------------------------

#!/bin/bash



#kill worker

ps axf | grep worker.py | grep -v grep | awk '{print "kill -15 " $1}' | sh



#kill master

ps axf | grep master_server.py | grep -v grep | awk '{print "kill -15 " $1}' | sh
```

修改 config 檔

遵照 How to train automatic speech recognition in kaldi 的教學檔中的第 5 點完成後會得到辨認器所需要的檔案，裡面有包含名為 conf 的資料夾，由於在生成時所給定的位置並不是在使用辨認器時的位置，因此我們需要修改它。

請將它修改成目前辨認器擺放的位置。

生成 yaml 檔

每次啟動辨認器時都會根據 yaml 檔裡的基本設定來開啟，複製以下內容生成檔案，但請修改路徑位置(黃色標部分)

```
nano formasa.yaml

---------------------------------------------------------------------------------

use-nnet2: True

decoder:

    nnet-mode: 3

    use-threaded-decoder:  true

    model : /home/brian/ASR-sysytem/kaldi_models/nnet_online/final.mdl

    word-syms : /home/brian/ASR-sysytem/kaldi_models/nnet_online/words.txt

    fst : /home/brian/ASR-sysytem/kaldi_models/nnet_online/HCLG.fst

    mfcc-config : /home/brian/ASR-sysytem/kaldi_models/nnet_online/conf/mfcc.conf

    ivector-extraction-config : /home/brian/ASR-sysytem/kaldi_models/nnet_online/conf/ivector_
extractor.conf

    max-active: 7000

    beam: 10.0

    frame-subsampling-factor: 3

    lattice-beam: 8.0

    acoustic-scale: 0.083

    do-endpointing : false

    extra-left-context-initial : 0

    min-active : 200

    acoustic-scale : 1.0

    endpoint-silence-phones : "1:2:3:4:5:6:7:8:9:10"

    traceback-period-in-secs: 0.25

    chunk-length-in-secs: 0.25

    num-nbest: 10

    add-pitch: true

out-dir: tmp



use-vad: False silence-timeout: 1000
```

```
post-processor: perl -npe 'BEGIN {use IO::Handle; STDOUT->autoflush(1);} s/(.*)/\1./;'

full-post-processor: /home/brian/ASR-sysytem/kaldi-gstreamer-server/sample_full_post_processor.py

logging:
    version : 1
    disable_existing_loggers: False
    formatters:
        simpleFormater:
            format: '%(asctime)s - %(levelname)7s: %(name)10s: %(message)s'
            datefmt: '%Y-%m-%d %H:%M:%S'
    handlers:
        console:
            class: logging.StreamHandler
            formatter: simpleFormater
            level: DEBUG
    root:
        level: DEBUG
        handlers: [console]
```

其中 out-dir: tmp 的設定，會將使用者傳送到 server 的音頻存在 tmp 的目錄之中，但是會轉成.raw 的格式。