Kaldi Extension

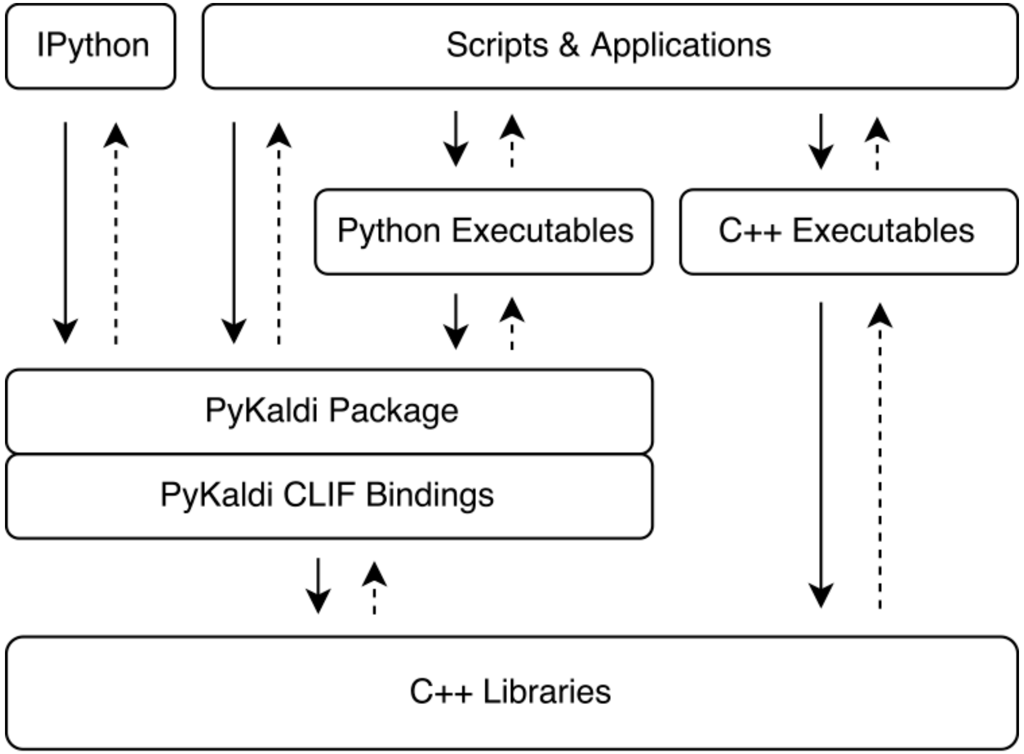
Yuan-Fu Liao

National Taipei University of Technology

PyKaldi

- PyKaldi is a Python wrapper for Kaldi. It aims to bridge the gap between Kaldi and all the nice things Python has to offer.
  - Near-complete coverage of Kaldi C++ API
  - First class support for Kaldi and OpenFst types in Python
  - Extensible design
  - Open license
  - Extensive documentation
  - Thorough testing
  - Example scripts
  - Support for both Python 2.7 and 3.5+

Architecture & Coverage



IPython

Scripts & Applications

Python Executables

C++ Executables

PyKaldi Package

PyKaldi CLIF Bindings

C++ Libraries

Package	Wrapped?	Pythonic?	Documentation?	Tests?
base				
chain				
cudamatrix				
decoder				
feat				
fstext				
gmm				
hmm				
ivector				
kws				
lat				
lm				
matrix				
nnet3				
online2				
rnnlm				
sgmm2				
tfrnnlm				
transform				
tree				
util				

Usage

```
# Define feature pipelines as Kaldi rspecifiers
feats_rspec = (
    "ark:compute-mfcc-feats --config=conf/mfcc_hires.conf scp:data/test/wav.scp ark:- |"
)
ivectors_rspec = (
    "ark:compute-mfcc-feats --config=conf/mfcc_hires.conf scp:data/test/wav.scp ark:- |"
    "ivector-extract-online2 --config=conf/ivector_extractor.conf ark:data/test/spk2utt ark:- ark:- |"
)
```

```
# Construct recognizer
decoder_opts = LatticeFasterDecoderOptions()
decoder_opts.beam = 13
decoder_opts.max_active = 7000
decodable_opts = NnetSimpleComputationOptions()
decodable_opts.acoustic_scale = 1.0
decodable_opts.frame_subsampling_factor = 3
decodable_opts.frames_per_chunk = 150

asr = NnetLatticeFasterRecognizer.from_files(
    "exp/tdnn_7b_chain_online/final.mdl",
    "exp/tdnn_7b_chain_online/graph_pp/HCLG.fst",
    "data/lang/words.txt",
    decoder_opts=decoder_opts,
    decodable_opts=decodable_opts)
```

```
# Decode wav files
with SequentialMatrixReader(feats_rspec) as f, \
     SequentialMatrixReader(ivectors_rspec) as i, \
     open("out/test/decode.out", "w") as o:
    for (key, feats), (_, ivectors) in zip(f, i):
        out = asr.decode((feats, ivectors))
        print(key, out["text"], file=o)
```

CF: Commandline

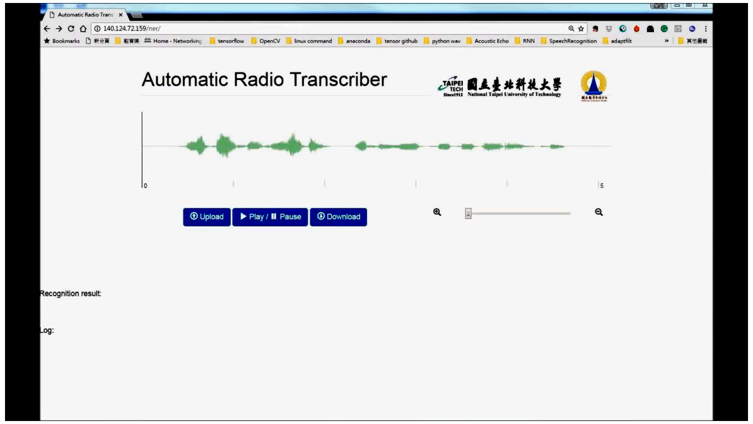
```
online2-wav-nnet3-latgen-faster --config=nnet_online/conf/online.conf
--add-pitch=true --do-endpointing=false --frames-per-chunk=20
--extra-left-context-initial=0
--online=true --frame-subsampling-factor=3
--min-active=200 --max-active=7000 --beam=15.0
--lattice-beam=6.0 --acoustic-scale=1.0
--word-symbol-table=nnet_online/words.txt
nnet_online/final.mdl
nnet_online/HCLG.fst
'ark:echo utter1 utter1|’ 'scp:echo utter1 C0000001.wav|’ ark:/dev/null
```

kaldi-gstreamer-server

- A real-time full-duplex speech recognition server
  - Implemented in Python
  - Kaldi toolkit
  - GStreamer framework

- Requirement
  - Kaldi
    - https://github.com/kaldi-asr/kaldi
  - kaldi-gstreamer-server
    - https://github.com/alumae/kaldi-gstreamer-server
  - gst-kaldi-nnet2-online
    - https://github.com/alumae/gst-kaldi-nnet2-online

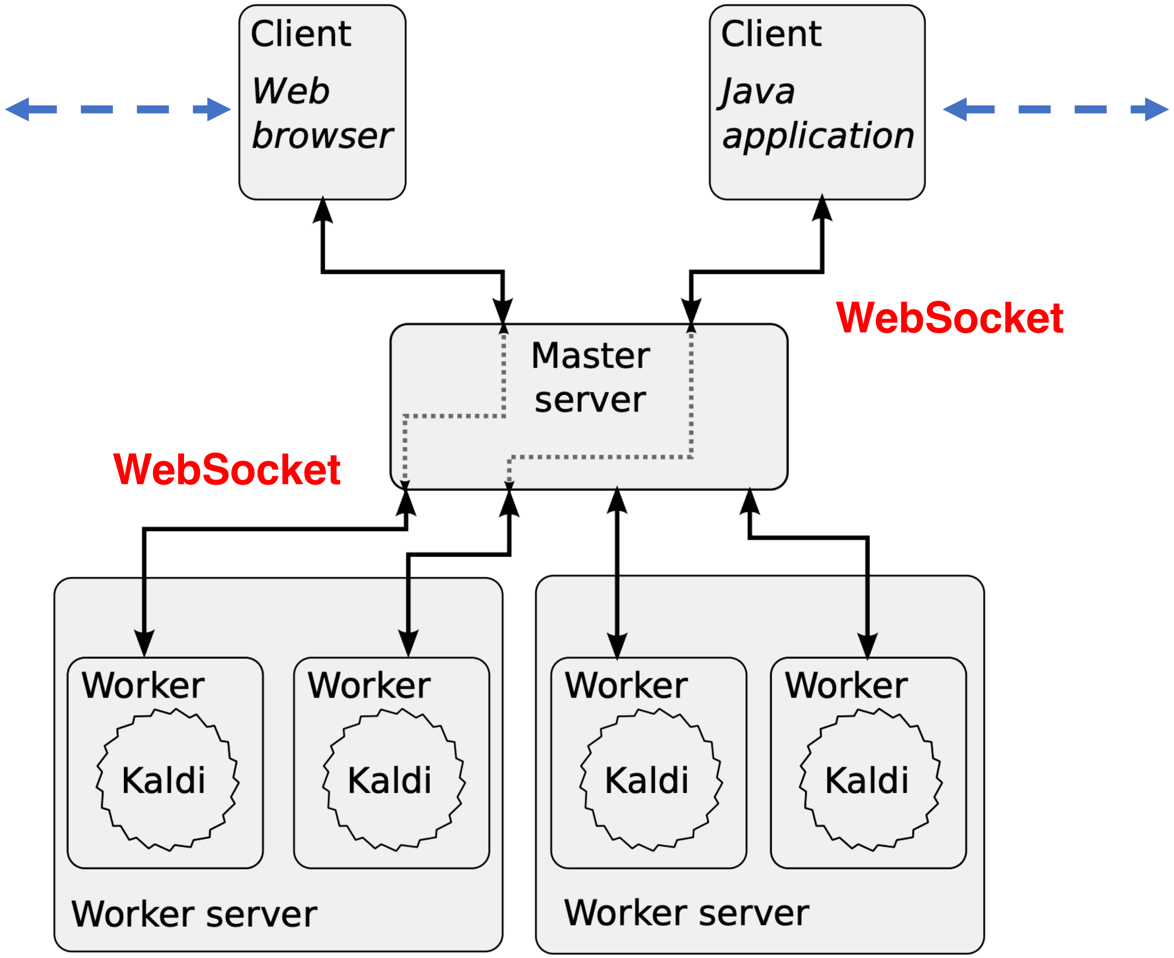
System Architecture



Automatic Radio Transcriber

Recognition result:

Log:



Client
Web browser

Client
Java application

WebSocket

Master server

WebSocket

Worker
Kaldi

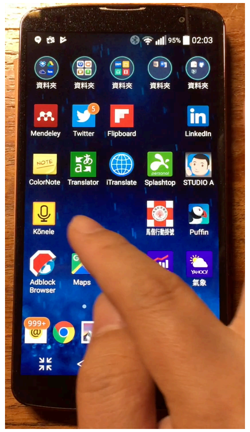
Worker
Kaldi

Worker
Kaldi

Worker
Kaldi

Worker server

Worker server



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_processo r.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]
```

Usage

- Running the master server

  ```
  python kaldigstserver/master_server.py --port=8888
  ```

- Running workers

  ```
  python kaldigstserver/worker.py -u ws://localhost:8888/worker/ws/speech -c sample_worker.yaml
  ```

- Server usage

  ```
  python kaldigstserver/client.py -r 32000 test/data/english_test.raw
  ```