Step-by-Step User Guide
Prototypical ASR in Upper Sorbian

This user guide describes all necessary steps to create and evaluate an adapted acoustic model for the prototypical Automatic Speech Recognition demonstrator in Upper Sorbian (HSB-ASR).

Software Installation

To recreate the models for the HSB-ASR demonstrator, the following software packages are required:

1. **dLabPro**, signal processing, and acoustic pattern recognition toolbox, and
2. **Unified Approach to Signal Synthesis and Recognition (UASR)**.

dLabPro and UASR are jointly developed at:

- TU Dresden at the Institute of Acoustics and Speech Communications (IAS)
  https://tu-dresden.de/ing/elektrotechnik/ias
- Lehrstuhl Kommunikationstechnik, BTU Cottbus
  https://www.b-tu.de/fg-kommunikationstechnik
- Fraunhofer IKTS-MD Dresden
  https://www.dresden.fraunhofer.de/de/institutes/fraunhofer_ikts.html

dLabPro

For more detailed information visit the Wiki page on:
https://github.com/matthias-wolff/dLabPro/wiki.

Installation on Linux

The installation of the dLabPro binaries under Linux operating system is quite straightforward, follow the instructions given here:

- https://github.com/matthias-wolff/dLabPro/wiki/Command-Line-Installation

For the installation of the python wrappers, the paths to the NumPy libraries and packages need to be set in the script:

```
programs/python/setup.py.
```

This could introduce problems related to different python versions on different platforms/OSs.

Adjust the paths in the `setup.py` script to point to NumPy folder.

Installation on Windows (MSYS2)

1. Download and install MSYS2 (https://www.msys2.org/)
2. Set the environment variables for MSYS.
   Add to PATH: C:\msys64\mingw64\bin
   Open the console with mingw64.exe and update the system with:

   ```
   pacman -Syu
   pacman -Su
   ```

3. Install toolchain for MinGW64:

```
pacman -S base-devel gcc make cmake vim mingw-w64-x86_64-gcc mingw-w64-
x86_64-ncurses mingw-w64-x86_64-readline
```

4. Start MinGW64 console.
5. Set environment variables:

```
export DLABPRO_HOME=<path_to_dLabPro_folder>

export UASR_HOME=<path_to_UASR>

cd $DLABPRO_HOME
```

6. Download the latest version of dLabPro from Git: https://codeload.github.com/matthias-wolff/dLabPro/zip/master

7. Unzip and enter the folder.
8. Make release version of dLabPro with:

```
make -sk -f Makefile RELEASE
```

UASR

UASR stands for Unified Approach to signal Synthesis and Recognition. The software implementation is a collection of scripts written in dLabPro jointly developed at BTU Cottbus-Senftenberg, Fraunhofer IKTS Dresden, and TU Dresden.

Download the repo and unzip in a folder parallel to “dLabPro”:

https://codeload.github.com/matthias-wolff/UASR/zip/master

For more detailed information about UASR, visit the Wiki page on https://github.com/matthias-wolff/UASR/wiki .

Speech Database

The speech database comprises features derived from speech recordings sessions (named as HSB) and the audio data obtained from the Upper Sorbian “Common Voice” dataset (named as CV).

Feature analysis is performed over the speech signals and the UPFA features calculated.

The features can be found in the sub-folder:

```
common/fea/UPFA_30
```

HSB-ASR Metadata and Configuration Files

Unpack the archived UASR data (db-hsb-asr-exp.zip). The db-hsb-asr-exp folder structure outlook:

```
\---common
    +---fea
    |   \---UPFA_30
    |       +---CV
    |       +---HSB-1
    |       |   \---RECS
    |       |       +---0001
    |       |       +---0002
    |       |       +---0003
    |       |       +---....
    |       |       \---0011
    |       +---HSB-2
    |       |   \---RECS
    |       |       +---0001
    |       \---HSB-3
    |           \---RECS
    |               +---0001
    +---flists
    +---grm
    +---info
    +---lab
    |   +---CV
    |   +---HSB-1
    |   |   \---RECS
    |   |       +---0001
    |   |       +---0002
    |   |       +---0003
    |   |       +---....
    |   |        \---0013
    |   +---HSB-2
    |   |   \---RECS
    |   |       +---0001
    |   \---HSB-3
    |       \---RECS
    |           +---0001
    +---lab_fsg
    |   +---HSB-1
    |   |   \---RECS
    |   |       +---0001
    |   |       +---0002
    |   |       +---0003
    |   |       +---....
    |   |       \---0013
    |   +---HSB-2
    |   |   \---RECS
    |   |       +---0001
    |   \---HSB-3
    |       \---RECS
    |           +---0001
    +---lexicon
    \---model
```

• fea	contains the pre-calculated features
• flists	file lists of the speech utterances, divided according to the domain and usage
• grm	Finite-State-Grammars for evaluation and with semantic tags
• info	configuration files for acoustic model adaptation and FSG evaluation
• lab	transliterations (word labels) for FSG evaluation
• lab_fsg	transliterations (word labels) for FSG evaluation with numbers as digits
• lexicon	HSB_ALL.lex is the default pronunciation lexicon
• doc	specifications and documentation
• model	default acoustic models

Acoustic Model Adaptation and Finite-State-Grammar Evaluation

1. Included pre-trained models:

`3_20.hmm`	`default acoustic model trained on German`
`3_20_hsb.hmm`	`3_20 model with removed non-HSB phonemes`

2. Model adaptation (adapt.cfg and adapt_hsb.cfg) with Windows CLI:

```
cd db-hsb-asr-exp
dlabpro ..\..\uasr\scripts\dlabpro\HMM.xtp adp common\info\adapt.cfg
dlabpro ..\..\uasr\scripts\dlabpro\HMM.xtp adp common\info\adapt_hsb.cfg
```

3. Model evaluation (eval.cfg and eval_hsb.cfg) with Windows CLI:

```
cd db-hsb-asr-exp
dlabpro ..\..\uasr\scripts\dlabpro\HMM.xtp evl common\info\eval.cfg
dlabpro ..\..\uasr\scripts\dlabpro\HMM.xtp evl common\info\eval_hsb.cfg
```

Speech Corpus Processing

The following python scripts (version >= 3.7) were for corpus preparation and analysis (in./common/scripts):

- `textSelect.py` takes UTF-8 textual files and generates phonetically rich sentences in as output in one or more datasets.
  - Input files:

sim.corpus	– used to set phonetic statistics (statistic corpus)
sim.vocab	– list of allowed words
HSB_ALL.txt	– corpus to select phonetically rich sentences
HSB_ALL.vocab	– vocabulary of valid words
phonmap.txt	– knowledge-based phoneme mapping

  - Output files:

textSelection.log	– log file with results
stats.pickle	– calculated statistics, if non existing they will be calculated from the statistics corpus

    - list of audio ids “sent/audio_diphones_<scoring>_<inventory>_<set>.txt”
    - sentences “sent/sentsel_diphones_<scoring>_<inventory>_<set>.txt”
    - pronunciation “sent/uttsel_diphones_<scoring>_<inventory>_<set>.txt”
  - Configuration parameters (beginning of the source code):

Parameter with default value	Description
mode = 'uasr'	“uasr” or “sampa“ phoneme inventory
split = 1	number of generated datasets
offset = 0	audio id starting number
num_sentences = 300	target sentence number
names = ["phones", "diphones", "triphones"]	phonetic category names
scoring_type = 3	Sentence scoring type (1, 2 or 3) 1 - each phonetic type should be present at least once 2 - negative log-prob weights 3 - sentence weights
basic_type = 'diphones'	basic phonetic type used for scoring

- Example usage:

  ```
  cd db-hsb-asr-exp\common\scripts\

  python textSelection.py   corpus\sim.corpus corpus\sim.vocab
        corpus\HSB_ALL.txt corpus\HSB_ALL.vocab phonology\phonmap.txt
  ```

- `corpusProcess.py` takes the result file from the previous script and creates normalized corpus, the lexicons and transliteration files.
  - Input files:
    - HSB_ALL.txt – non-normalized corpus
    - phonmap.txt – knowledge-based phoneme mapping
  - Output files:
    - HSB_ALL.lex – pronounciation lexicon
    - HSB_ALL.corpus – normalized corpus
  - Example usage:

    ```
    cd db-hsb-asr-exp\common\scripts\

    python corpusProcess.py corpus\HSB_ALL.txt phonology\phonmap.txt
    ```

Evaluation of New Speech Recordings on the SMARTLAMP Domain

1. Create a folder named “sig” under the folder “common”.
2. Copy speech recordings in the format:
   - Coding: Pulse Code Modulation (PCM)
   - File extension: wav
   - Channels: mono
   - Sample Rate 16 kHz
   - Resolution 16 bit
   - Byte Order: Little Endian

   into the “sig” folder.

3. Create a “test.flst” file into the “flists” folder,
   Add relative paths to the sig folder (for example see existing file lists)

4. Copy and rename the eval_hsb.cfg to eval_new.cfg, open and replace the line:
   uasr.flist.test = "test.flst"

5. Run evaluation with:

   ```
   dlabpro ..\..\uasr\scripts\dlabpro\HMM.xtp evl common\info\eval_new.cfg
   ```

   Note: BAS SpeechRecorder tool can be used for recording sessions:

   https://www.bas.uni-muenchen.de/Bas/software/speechrecorder