Appendix A

Contents of the included media

This manual lists contents of the included media:

- `sub_eval.zip` – Zip file with the "sub_eval" dataset used in the experiments, DL here: sub_eval.zip. Contains *eval_clear* – `wav` audio data without pre-recorded messages and *eval_goal* – `wav` audio data with mixed pre-recorded messages. The subfolders include processed phoneme posteriors, bottleneck features in `htk` format, and phoneme strings in `txt` format.
- `messages/` – Original unmixed messages with transcription.
- `scripts/` – Created and used source codes.
  The folder `scripts/` contains these subdirectiories:
  - `scripts/evaluations/` – The scores, processing time and created objects (persistent Python objects (RQA list, cluster analysis) in `pkl` format) from the experiments.
  - `scripts/third_party_scripts/` – The folder contains third party source codes.
- `text/` – LaTeXsource codes of the paper.
- `Excel_BobosDominik.pdf` – This paper in the PDF format.

Appendix B

Manual

This manual describes installation process and user manual. The setup is tested, and it is working on Ubuntu 20.04 machines.

B.1 Installation manual

To have properly working scripts, it is necessary to install Python dependencies. Installation can be done using the following commands:

```
cd scripts/
pip install -r requirements.txt
```

Scripts for editing audio files need `ffmpeg` utility. This dependency can be installed by `sudo apt-get install ffmpeg`.

Additionally, in the case of creating phoneme posteriors, it is necessary to install the phoneme recogniser, from https://speech.fit.vutbr.cz/software/phoneme-recognizer-based-long-temporal-context. After successful installation, change path to installed phoneme recogniser in script `/scripts/set_phnrec`.

B.2 User manual

Creation of a simulated dataset

The creation of a simulated dataset is accomplished by two scripts `split.py` and `mix.py`.

Script split.py

Python script *split.py* is responsible for cutting phone calls into smaller parts.

There are many arguments to specify wanted result:

- `--sec` – Sets the length of the cut part (in seconds).
- `--src` – Path to source directory containing the uncut phone calls.
- `--dst` – Path to destination directory where the cut phone calls will be saved.
- `--lt, --st` – Will cut longer/shorter recordings than specified (in seconds).

- `--rm` – Remove original file after splitting.
- `--stats` – Shows statistics about the files from given source.
- `--move, --copy, --count` – Will move/copy "count" recording.

Example command for splitting files to 180-second-long segments:

```
python3 split.py --src=/source_folder/ --dst=/destination_folder/ --sec=180
--lt=100 --st=500 --rm
```

Script mix.py

Python script *mix.py* is responsible for mixing pre-recorded telephone operator messages into phone calls.

First, the given message is edited in repetitions, speed, volume, and afterwards, it is inserted into the phone calls. Arguments to specify wanted result include:

- `-m, --mode` – Specifies, which type of pre-recorded message you want to use (possible values are: A, B, C).
- `--mpath` – Path to the directory with the pre-recorded messages.
- `--spath` – Path to the directory with the speech .wav files.
- `--export` – Path to the directory to export the mixed audio files.
- `--lt, --st` – Will use longer/shorter recordings than specified (in seconds).
- `-g, -s, -r` – Change volume/speed/repetitions of the pre-recorded messages.
- `--random` – severity of randomness. Possible values are:
  - *0* – no random values, use the original values of the pre-recorded message
  - *1* – low differences (gain between -3dB+3dB, speed 0.95-1.05, repeat 1-10.0)
  - *2* – optimal differences (gain between -6dB+6dB, speed 0.9-1.1, repeat 0.8-30.0)
  - *3* – high differences (gain between -10dB+6dB, speed 0.85-1.15, repeat 0.7-40.0)
- `--onlymodify` – Will modify the operator message only, without mixing into phone calls.
- `-stats` – shows statistics about the files from given source

Example command for mixing pre-recorded messages of type A to audio recordings:

```
python3 mix.py --mpath=/messages/ --spath=/telephone_conversations/
--export=/mixed_conversations/ -m=A --lt=100 --st=150 --random=2 -g -s -r
```

Experiments

This section provides detailed instructions on how to obtain the results presented in Chapters 5, 7 and 8. The process can be described as follows:

To start the classification process, it is necessary to set a source directory containing goal (with mixed messages) and clear (pure phone calls) data. This is accomplished by `--src` argument.

It is necessary to specify the system to classification by the `--system` argument. The options are as follows:

- Baseline DTW system (in 5.1) – `--system=basedtw`
- Baseline RQA system (in 5.2) – `--system=rqa_unknown`
- Baseline FSM system (in 5.3) – `--system=fuzzy_match_base`
- RQA+SDTW clustering+FSM system (The final system)
  - unknown messages scenario: `--system=rqacluster_fuzzy_match_unknown`
  - known messages scenario: `--system=rqacluster_fuzzy_match_known`

To be clear, the final system is using the RQA list of candidates and requires the file *evaluations/objects/rqa_list_FEAT.pkl*, where "FEAT" is the one specified in *--feature* argument. If the file is not present, the system will create one. The same applies for S-DTW clusters, where the system looks for the file *evaluations/objects/cluster_rqa_list_CLUSTFEAT.pkl*, where "CLUSTFEAT" is the feature specified by *--cluster-feature*. It is required to specify used type of feature arrays:

- RQA, DTW system – `--feature=mfcc/posteriors/bottleneck`
- FSM system – `--feature=string`
- The final system needs additional parameter – `--cluster-feature=mfcc/posteriors/bottleneck`

The systems using frame averaging are specified with `--frame-reduction=NUM`.
Parallelisation is accomplished by `--parallelize-from` and `--parallelize-to` parameters. The parameters specify which files are processed by the system.
The example command can be as follows:

```
python3 main.py --src=../sub_eval/ --system=rqacluster_fuzzy_match_unknown
--feature=string --cluster-feature=bottleneck --parallelize-to=200
```