Interaction Style Modeling Toolset

an extention of the Midlevel Prosodic Features Toolikit

Version 1.0

Nigel Ward, University of Texas at El Paso

July 28, 2020

1 Overview

This toolset supports the analysis of interaction styles in spoken dialog. To quote the abstract of my as-yet-published paper:

> In spoken dialog, people clearly vary in their interaction styles, but a comprehensive model of the space of variation has been lacking. To address this need, I applied Principal Component Analysis, using features designed to capture aspects of interaction-related prosodic behaviors, to 32084 conversation fragments from the Switchboard corpus of American English telephone speech. This gave 8 meaningful dimensions of variation, including some not previously note in the literature. The resulting vector space representation, and the discovery method, may be useful for dialog systems design, customization, and adaptation.

This toolset has so far been used only for the analyses reported in the paper, but the intent is to generally support computational analyses of interaction styles. Sample applications are discussed in the paper.

Specifically, this code enables you to:

- derive a vector-space representation of interaction styles from a corpus of stereo-recorded spoken dialogs
- given any new dialog, compute where it lies in an existing space of interaction styles
- given a corpus with metadata, such as Switchboard, compute various correlations and statistics on the factors affecting interaction styles

This document is a work in progress. Comments and suggestions are welcome.

2 Getting the Code

This code was written in Matlab and runs on Matlab version 2019.

The code is at `https://github.com/nigelgward/TBD` . It requires the Midlevel Toolkit, which is available at `https://github.com/nigelgward/midlevel/` , complete with documentation.

3 Terminology

The term "dimension" here sometimes refers to prosodic construction dimension and sometimes to interaction style dimensions.

4 Corpus Preparation

Create a directory somewhere with lots of space, refered to below as `motherDirectory`, and copy the Switchboard data discs there one by one, naming the subdirectories `disc1` ... `disc4`.

Create `wav`-format copies of each audio file using `code/sph-to-wav.sh` .

Create the pitch files using `midlevel/src/reaperize.sh` .

Decide how to split the data into subsets, and create an index file for each How this was done for Switchboard is described in `labnotes.txt`. Each index file contains a one-per-line listing of the audio files in that set.

5 Building the Model

Open Matlab and change to the `stats` subdirectory..

Run `computeStyleParams.m('stats/trainset.txt','trainStats.csv')` to create the features for all files in the training set. The output, here `trainStats.csv`, is sometimes called a PCB file, as it contains features based on prosodic-construction bin counts.

This will take many hours, so if it crashes, be prepared to look at `trainStats.csv` to see how far it got, and edit the code to just append stats for the files not already processed.

This file has three hidden dependencies, reading data from hardcoded filenames. The first two, `fsfile` and `rsfile` specify how to compute the prosodic dimensions, as described in the midlevel documentation. The third is `sifile` is used to specify the bins. Usually this should be held constant for all analyses, unless you want to define not only a new interaction space but also to redefine the basic features used.

Now run `deriveISspace('trainStats.csv', true, 'trainIsNormRot.mat', 'trainIStyles')`. This will create the model and save its parameters in the `mat` file. It will also print out many interesting statistics that can be copied into the paper. It also pops up a figure that will go in the paper (when compareWithSubsets) is uncommented.

will also write several files, notably:

- `loadingsTables.txt`, written to `stats` shows, for each dimension (each interaction-style dimension) the loadings on the behavioral features (the prosodic-constructions-binned features).
- many files with names starting with `isratios`. These can be examined, for example by `sort -n isratios1hi.txt | more`. Once computed, it's nice to save them in a safe place, for example by moving them all to `wordstats`.
- many files with names starting with `idimWords`. These can be examined with LIWC.
- `sox-commands.sh`, a script which can be run with `bash` to generate various audio fragments. These can then be listened to to help understand the meaning of the various dimensions. Once created, it's nice to move them somewhere safe, e.g. `exemplars`. The script also creates some anchor stimuli, to use in the human-perceptions experiment.

6 Applying a Model to New Data

This is to be done, for example, when choosing clips to use for the experiment, which should, of course, be taken from data unseen in training the model or interpreting the dimensions: thus from held out testset data.

** not tested yet **

First `computeStyleParams.m('code/testset2.txt','test2Stats.csv')`.

Now run `deriveISspace('test2Stats.csv', false, 'trainIsNormRot.mat')`, importantly specifying `false` for the second parameter.

7 Acknowledgments