Appendix A

Pipeline User Guide

This guide intends to provide guidance to the users of this pipeline. This tool provides a command-line interface allowing users to join data from multiple devices and different formats, focusing on data from driving simulators. The tool containing the pipeline is launched in the command line by running the script through the command 'python configuration.py'. To perform this action, it is necessary to be in the directory that contains the Python file. The program has three main options:

a) Extracting and fusing files from the BBAI experiments;

b) Extracting and fusing files from other experiments;

c) Mapping features of datasets from other experiments.

A.1 Option A

The first option automatically fuses and preprocesses the data exactly as it is extracted from the devices used in the BBAI experiment. When selecting this option, it is asked to indicate the experiment context: drowsiness or distraction. Once this choice has been made, a list of options containing the possible files to extract is presented, as seen in figure A.1. After the user chooses the desired combination of files, it is presented with a file explorer window in order to select the folder that contains to be extracted and fused.

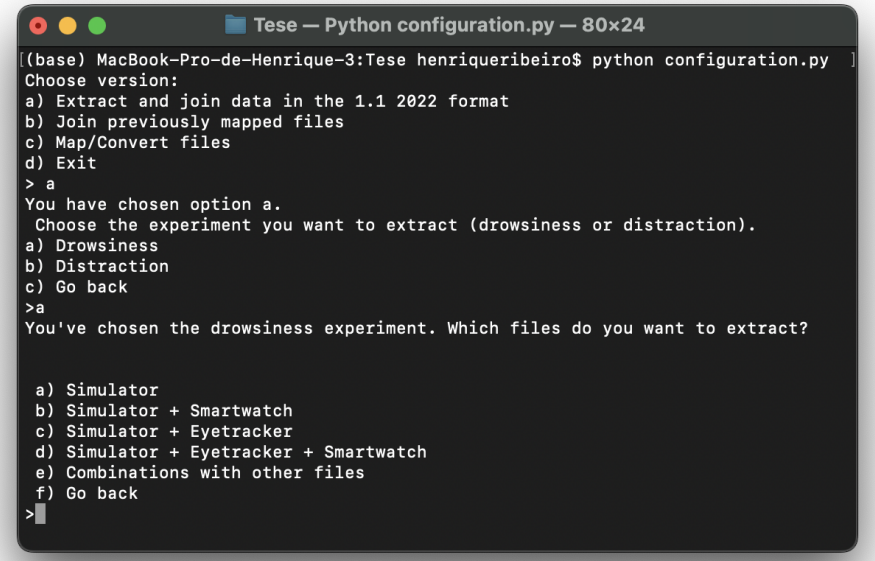

Figure A.1: Option A - Possible combinations of devices

Since the names and directories of the files are not always the same, the directories where the files are located must be in one (or more) of the formats listed below. In the directory formats listed below, the variable *base_path* represents the directory or folder chosen by the user, *i* denotes the participant's ID, and *type* indicates the type of gaze files, which may be 'Left', 'Right', or 'Vergence'. The asterisk (*) is used as a wildcard character to denote any sequence of characters that may appear in that part of the directory structure. The variable *exp* represents the experiment and can be either 'dist' for distraction data or 'sono' for drowsiness data.

Eyetracker - Possible directories:

- `{base_path}/{i}/{exp}_{i}_*_*_*/{exp}_{i}_eyetracker_*_*_*/*/*/* Gaze-{type}.csv`
- `{base_path}/{i}/{exp}_{i}_*_*_*/{exp}_{i}_eyetracker_*_*_*/*/* Gaze-{type}.csv`
- `{base_path}/{i}/{exp}_{i}_*_*_*/{exp}_{i}_eyetracker_*_*_*/*/Outputs/* Gaze-{type}.csv`
- `{base_path}/{i}/{exp}_{i}_*_*_*/{exp}_{i}_eyetracker_*_*_*/* Gaze-{type}.csv`

Smartwatch - Possible directories:

- `{base_path}/{i}/{exp}_{i}_*_*_*/{exp}_{i}_smartwatch_*_*_*/*.csv`

Simulator .log Files - Possible directories:

- `{base_path}/{i}/{exp_}{i}_*_*_*/{exp_}{i}_simulador/{exp_}{i}_simulador_telnav.log`
- `{base_path}/{i}/{exp_}{i}_*_*_*/{exp_}{i}_simulador/{exp_}{i}_simulador.log`

Simulator .txt Files - Possible directories:

- `{base_path}/{i}/{exp_}{i}_*_*_*/{exp_}{i}_simulador/{exp_}{i}_simulador_txt.txt`
- `{base_path}/{i}/{exp_}{i}_*_*_*/{exp_}{i}_simulador/{exp_}{i}_simulador.txt`

Once the folder containing the files has been chosen, the program fuses the files and asks for the folder where the files shall be saved.

If the user, when prompted to select the combination of desired files chooses the last option ("e) Combinations with other files") after the process described, above a new step is added. Even with this option, the only strictly necessary files are the ones related to the simulator. The user is prompted if he would like to select a single file or a folder containing multiple files. The program then opens a file explorer window that lets the user select the file/folder containing generic data to the already existing fused dataframe. However, this is only possible if the newly added files contain a feature called 'tempo (s)', representative of the timestamp, and other called 'id' , which is the identification number of the participant. It is essential to note that in instances where the user opts to select a folder containing generic data, only the initial file corresponding to a repeating participant identifier (ID) will be integrated. The selection of the folder is designed to amalgamate multiple files from the same device across different participants, rather than to compile files from multiple devices for a single user. In this option, the algorithm performs several enhancements to the data, such as, missing value imputation, outliers treatment and feature engineering.

A.2 Option B

This option allows the user to merge multiple, much like the first option does, however, this does not follow the structure of files present in the BBAI version. In this regard, instead of selecting only a folder and the algorithm "finding" where the relevant files are, the user needs to select the files or folder containing them regarding each device. After selecting the desired combination of devices, the user is then prompted to select each file/folder individually, as aforementioned. After selecting, for example, the combination of Simulator and Smartwatch, the user is asked whether he would like to select a file or a folder, and, once he has done this a window opens with a prompt asking the user to select the file/folder containing the data related to the vehicle simulator (velocity, etc.) (as seen on figure A.2 , followed by a prompt asking for the data related

to the participant and interaction with vehicle IVIS system (ID, KSS, etc.) and finally the data related to the smartwatch. To successfully merge files using this method, the features within the files must follow certain naming conventions. These naming conventions are presented before prompting the user with the files that have this requirement. To solve this problem, option c) is provided in the program. It is also necessary that the files containing this data are in .csv format, using a semicolon (';') as a separator.

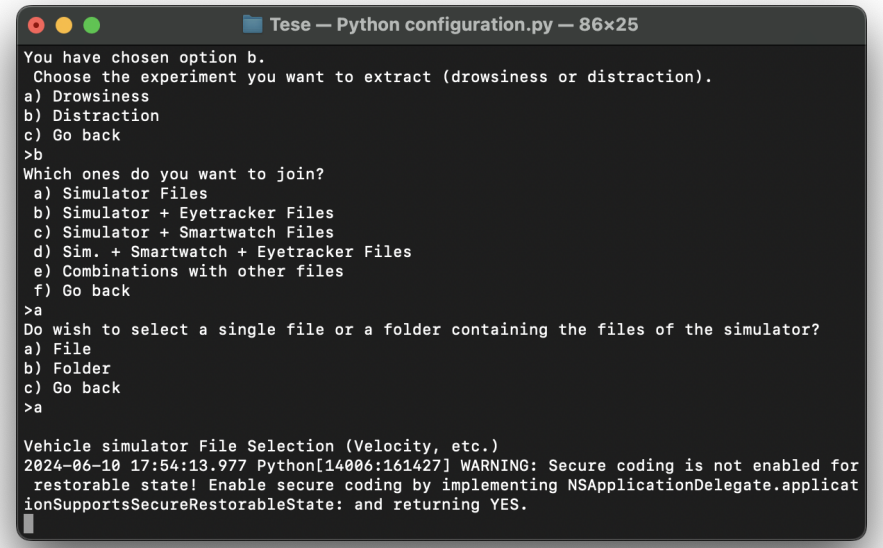

Figure A.2: Option B - File choice

A.3 Option C

This option allows the user to map certain features to the denominations needed to merge the data in option B. This option, once again, prompts the user to select the experiment, the device whose features he wants to map, and, if he wants to select a file or a folder containing a larger number of files. Once the user selects his preferences and the directory of the files/folder, he is asked whether he wants to perform the mapping manually or through a preconfigured file. If the manual option is chosen, the user is asked to map the features manually, as seen on figure A.3. Here, the participant must input the names of the features of the original files that correspond to

the ones that appear on the interface. He may skip the features that are not useful in relation to their input data by writing 'skip' or pressing enter. Nonetheless, some features are mandatory, and when trying to skip, the program will force the user to do so. After manually mapping the features, a prompt appears asking the user whether he may want to add additional features contained in the input files to the output file. By writing 'yes' the user can write the names of the desired features and subsequently they will be added to the output files. If the user opts to map the features using a pre-configured file, they will be prompted to select the appropriate file. This file must be in .json format, similar to the one depicted in the listing A.1, with mappings located on the right side of the file. Upon uploading the .json file to the program, the mappings will be executed automatically. Following this, the user will receive a prompt to save the newly configured files.

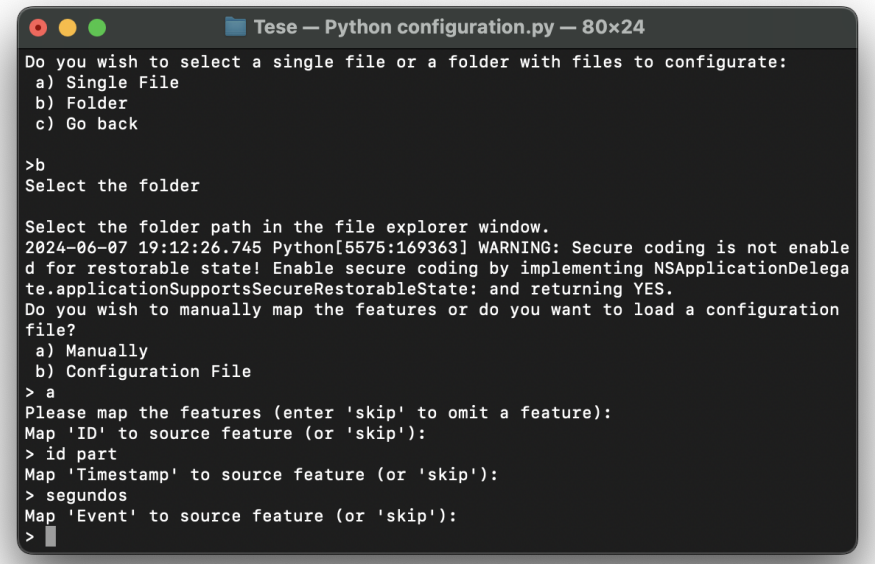

Figure A.3: Option C - Feature mapping

```
1 {
2 "column_mapping": {
3 "ID": {"mandatory": true},
```

```
4 "Timestamp": {"mandatory": true},
5 "KSS": {"mandatory": true}
6 }
```

Listing A.1: Pre-configured JSON file

We hope this guide has provided you with all the necessary information to utilize our tool effectively. For further assistance contact henriquebribeiro00@gmail.com.