Dokumentacja systemu do liczenia ivectorów oraz REST-owego API.

Python: 2.7.9

Potrzebne biblioteki znajdują się w pliku requirements.txt

I

W skrypcie wave2ivec.py znajduje się klasa **IVector** wraz z potrzebnymi metodami (niektóre są tylko konieczne do obliczeń)

IVector:

__init__() : wczytuje potrzebne modele oraz pliki na stałe do pamięci RAM

wav_conversion(y, fs) : resampluje plik wav do 8000Hz, przyjmuje wav w próbkach oraz częstotliwość próbkowania; zwraca zresamplowany wektor oraz nową częstotliwość próbkowania

proces_wav(wav_file, mode="ivector", vad_dir="auto") : przyjmuje plik .wav, vad_dir jest ustawiony na "auto", aby liczył się automatycznie, funkcja posiada 3 warianty:

- mode="ivector" : return ivector -> zwraca ivector w postaci wektora
- mode="mfcc" : return fea -> zwraca mfcc dla nagrania w postaci macierzy
- mode="statistics" " return f, n -> zwraca statystyki: n – zero order, f – first order

save_ivector_to_file(ivector, path) : przyjmuje ivector oraz scieżkę, zapisuje ivector do pliku tekstowego

mfcc_to_ivector(fea) : przyjmuje macierz mfcc, zwraca ivector

get_matrices(mode="") : przyjmuje argument, zwraca gotową macierz dla wytrenowanego modelu: mode =

- „tv"
- „lda"
- „plda_c"
- „plda_gamma"
- "plda_k"
- "plda_lambda"

get_score_from_ivectors(path) : przyjmuje scieżkę do folderu z ivectorami (format .ivec), zwraca scorew postaci wektora

get_score_from_ivectors_zip(zip_file) : przyjmuje pliki .ivec skompresowane w pliku .zip, zwraca score w postaci wektora

metody klasy IVector korzystają z innych metod zawartych w plikach .py w folderze projektu oraz z modeli w folderze „models"

II

client.py – przykładowe korzystanie z API na localhoscie

wysyłanie wav, otrzymywanie ivectora; wysyłanie parametru aby otrzymać macierz

III

biometric.py – kod API, korzysta z frameworku FLASK. Dzięki bibliotece flask_swagger_ui oraz dołączeniu pliku .json w folderze static, stworzono interaktywną dokumentację w swagerze (http://localhost:5000/swagger przy testowaniu na localhoscie)

API jest w konwencji REST: oznacza to, że posiada metody tj. GET, POST, PUT, DELETE. W stworzonym API zostały użyte metody GET oraz POST. GET zwraca wartość bez uploadowania żadnych danych. POST zwraca wartość procesując wysłane dane np. plik .wab lub string.