

Travel Tales Prototype Documentation

Abstract	2
Introduction to TravelTales: In-Car Conversation Recommendation System	3
State of the art for Conversation Engagement Monitoring	4
Architecture	5
Indexer	6
RSS Feeds and MORSS	6
Indexer Operations	7
News Embedding Model	8
Dataset	8
Model Architecture	8
PCA of the collection	9
Summarization Model	11
Text to Speech Model	12
Server	13
Registration	14
Retrieval System	15
Client	15
Data Visualization Module	16
Engagement Level Regressor	17
Speaker Recognition Module	18
Audio Sentiment Classification	19
Video Sentiment Classification	20
User Manual	21
A. User Registration	21
B. On-board passengers selection	22
C. Music pause/unpause	23
D. News player commands	24
E. Plots panels	25
F. Feedback Gathering (with camera)	26
G. Explicit feedback form	27
Conclusions	28
References	29

Abstract

As the automotive industry progresses towards enhanced connectivity and technological integration, the evolution of in-car entertainment systems introduces both opportunities and challenges. Our work addresses the general problem encompassing in-car entertainment, the dynamics of passenger interactions within the vehicle and potential distractions for drivers.

The increasing sophistication of in-car entertainment systems, often designed to offer a myriad of features, raises concerns about potential distractions for drivers. As vehicles become more connected, the temptation for drivers to engage with various entertainment options poses a risk to road safety.

Simultaneously, there exists a growing interest in fostering meaningful interactions among passengers during car journeys. Balancing the desire for engaging in-car experiences with the imperative of maintaining a safe driving environment is a critical challenge.

Our project proposes a solution for addressing the dual objectives of providing entertaining in-car experiences while mitigating distractions for drivers. The subsequent chapters delve into a comprehensive exploration of solutions, particularly emphasizing the development of intelligent recommendation systems.

Introduction to TravelTales: In-Car Conversation Recommendation System

TravelTales introduces an advanced in-car conversation recommendation system leveraging artificial intelligence, machine learning, face and voice recognition technologies. The primary objective is to enhance the overall driving experience and stimulate engagement among vehicle occupants.

The system employs machine learning to continuously analyze in-car conversations, creating individual interest profiles for each occupant. This data-driven approach aims to offer personalized conversation prompts aligned with the unique preferences of passengers.

The inclusion of a robust voice recognition system allows TravelTales to automatically identify distinct occupants within the vehicle, enabling a tailored experience for each individual based on their preferences.

Key features include personalized recommendations, where the system suggests conversation topics based on occupants' known interests. The flexibility in presentation modes allows occupants to receive recommendations through voice or text messages on the infotainment system.

Real-time updates ensure that occupants stay informed about the latest developments related to their identified interests, contributing to a dynamic and engaging in-car experience.

Benefits of TravelTales include an enriched travel experience with more engaging conversations, promotion of meaningful dialogues between the driver and passengers, and improved road safety by allowing occupants to remain focused on driving while receiving recommendations.

In summary, TravelTales is positioned as an innovative in-car conversation enhancement tool, utilizing technology to provide personalized and dynamic content tailored to the preferences of each occupant.

State of the art for Conversation Engagement Monitoring

In this chapter, we delve into the existing literature that forms the basis for our current project, drawing inspiration from the work titled **"A multimodal approach for modeling engagement in conversation."** Authored by F. Morreale, C. Cosi, L. Benech, E. Nisi, and V. Pallotta published in **Frontiers in Computational Neuroscience** in 2023 [1], this research introduces a new methodology for predicting engagement levels in conversations through a multimodal approach. Their method incorporates prosodic, mimo-gestural, and morpho-syntactic features to achieve a superior assessment of engagement compared to previous models, beating a state-of-the-art weighted F-score. The authors highlight the importance of combining modalities.

The paper suggests a new way to define engagement that works for both regular and online conversations. They put this new idea into action using a plan to mark two sets of videos showing people talking face-to-face. These marks cover lots of different ways people talk, like smiling, nodding, giving feedback, and how intense their hand movements are.

Beyond its theoretical contributions, the work holds practical implications for human-machine interaction. The ability to automatically assess engagement can significantly impact the design of socially-aware robots and conversational agents. For instance, a robot could dynamically adjust its behavior during a conversation based on engagement levels, contributing to more natural and responsive interactions.

This literature review lays the foundation for our project, introducing essential concepts and methods that shape our method. Our project in fact adopts a multimodal approach, integrating the audio signal sentiment, speech rate, speech duration, and user video sentiment to estimate engagement levels.

Architecture

The Hardware and Software Architecture of Travel Tales application is based on a Client-Server structure:

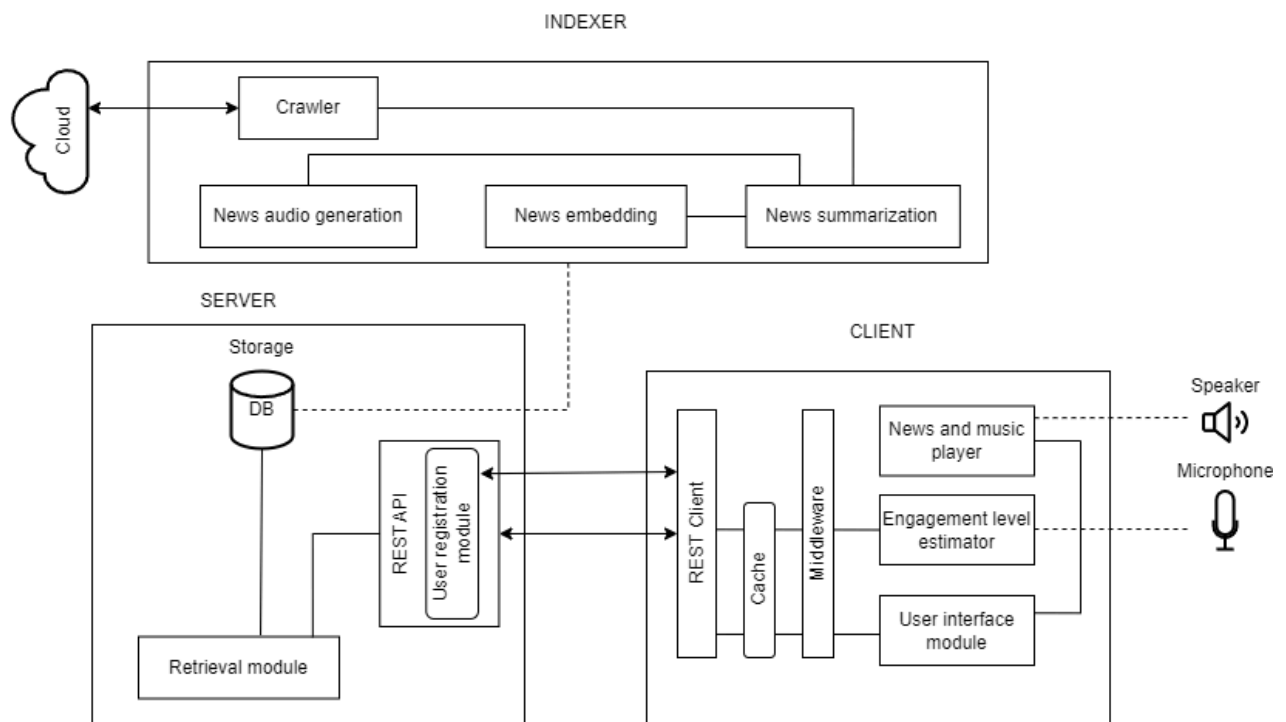
- **Client Side Implementation in Python:**

The application's user interface is crafted using Python, specifically using Python SimpleGUI. This choice ensures a user-friendly graphical interface and the flexibility of Python for client-side functionalities. The Client application runs on the **Raspberry Pi 4** to simulate the car environment.

- **Server Side Implementation using Python Flask:**

On the server side, the Travel Tales application is implemented in Python, with Flask serving as the framework to create and manage endpoints accessible by the client. Flask provides a solid foundation for handling requests and responses, ensuring a smooth flow of data between the client and server. The Server application runs on a **Laptop**, capable of providing more computational power to manage requests from different Clients.

- **Indexer** is a separate module, where the collection of News is created. It requires high computational power to run Neural Network models capable of creating news embeddings, news summaries and audio files from text. Such a module was developed using Jupyter Notebooks inside **Google Colab**, in order to use GPUs in the training and inference phases.



The **Client software architecture** is composed by the following modules:

1. **REST client module**, which is in charge of exposing the methods to query the server for news suggestions and to send to the service server the users engagement levels.
2. **Engagement Level Estimator module**, which is in charge of recording the conversation generated by news proposal and estimating the engagement level

3. **User Interface module & News and music player:** it reproduces the vocal news and the ambient music played by the system. It handles the speakers and exposes a visual interface to pause/unpause music and to switch proposed news.

The **Server software architecture** is composed by the following modules:

1. **REST APIs and user registration Module**, which manages the communication between the server and client devices. It exposes a web interface to allow new users registration, news suggestions, etc...
2. **Retrieval Module:** it exposes methods to retrieve the best news to suggest to a given group of users.

The **Indexer architecture** is composed by the following parts:

1. **News summarization:** it is in charge of distilling news article text into concise and informative summaries, using natural language processing techniques to extract key information.
2. **News embedding:** it is responsible for creating news embeddings as compact and meaningful representations and employs neural networks to transform textual information into a numerical format.
3. **News audio generation:** it generates audio files from news summaries, synthesizing real voices using AI techniques.

Indexer

RSS Feeds and MORSS

The **Indexer** component developed in Google Colab uses **RSS Feeds** as news article source, which is a web feed format used to publish frequently updated content, such as blog posts, news headlines, audio, and video, in a standardized way. The purpose of an RSS feed is to enable users to subscribe to their favorite websites or online content sources and receive updates in a centralized location.

A lot of newspapers only provide a small intro of their articles in their RSS feeds. As this is not really convenient, **morss.it** turns those shortened RSS feeds into feeds with full articles. In the following collection exploited RSS Feeds are listed:

```
rss_url_list =[
    "https://morss.it/clip/https://www.huffpost.com/section/us-news/feed",
    "https://morss.it/clip/feeds.bbc.co.uk/news/rss.xml",
    "https://morss.it/clip/https://www.huffpost.com/section/business/feed",
    "https://morss.it/clip/https://www.huffpost.com/section/celebrity/feed",
    "https://morss.it/clip/https://www.huffpost.com/section/media/feed",
    "https://morss.it/clip/https://www.huffpost.com/section/tv/feed",
    "https://morss.it/clip/https://www.huffpost.com/section/money/feed",
    "https://morss.it/clip/https://www.huffpost.com/section/worklife/feed",
    "https://morss.it/clip/https://www.theguardian.com/sport/blog/rss",
    "https://morss.it/clip/https://21sportsnews.com/feed/",
    "https://morss.it/clip/https://news.22bet.com/news/feed/",
    "https://morss.it/clip/https://www.247sportsgist.com/feed/",
    "https://morss.it/clip/https://ashsportstalk.org/feed/",
```

```

"https://morss.it/:clip/rss.cnn.com/rss/money_markets.rss",
"https://morss.it/:clip/rss.cnn.com/rss/money_pf_taxes.rss",
"https://morss.it/:clip/https://www.theverge.com/google/rss/index.xml",
"https://morss.it/:clip/https://www.wired.com/feed/tag/ai/latest/rss",
"https://morss.it/:clip/https://www.wired.com/feed/category/science/latest/rss",
"https://morss.it/:clip/https://nypost.com/entertainment/feed/",
"https://morss.it/:clip/https://nypost.com/media/feed/",
"https://morss.it/:clip/https://www.latimes.com/environment/rss2.0.xml",
"https://morss.it/:clip/https://www.etonline.com/news/rss",
"https://morss.it/:clip/https://therevealer.org/feed/",
"https://morss.it/:clip/https://thepoliticalinsider.com/feed/",
"https://morss.it/:clip/https://www.teachertoolkit.co.uk/category/podcasts/feed/",
"https://morss.it/:clip/https://humornama.com/feed/",
"https://morss.it/:clip/https://www.sciencedaily.com/rss/top/science.xml",
"https://morss.it/:clip/https://www.newscientist.com/subject/space/feed/"
]

```

This list of RSS Feeds was made to collect at least one news article for each category of **Embedder module**.

Indexer Operations

After a news article is retrieved by the Indexer, it filters the key fields like Title, Body, Thumbnail, ... and execute three main operations:

1. **Summarization**, which consists in summarizing the Body of the article.
2. **Embedding**, which consists in obtaining a vector of 14 dimensions, one for each category of the embedder, from the news Body input text.
3. **Audio generation**, which consists in generating a realistic and natural sounding audio track from the news article summarization, using a sampled voice.

In the following chapter more details about Neural Networks exploited are provided.

The Outputs of the Indexing stage are two:

1. A **collection of News samples**, where each record is composed by the following fields:
 - ID, id of the article
 - Link, link to the article on the web
 - Title, title of the article
 - Summary, summary of the article
 - Article, textual body of the article
 - Date, date of publication
 - Embedding, article text embedding
 - Wav-link, link to article wav file
 - wav_file_name, filename of the article wav file
2. A **collection of Audio tracks**, one for each news article summary.

	Link	Title	Summary	Article	Date	Embedding	Wav-link	wav_file_name
0	https://www.huffpost.com/entry/federal-judge-tanya-chutkan-trump-case-swatting_n_659c69cf4b0f9f621dec83	Judge In Trump's Election Interference Case Reportedly Targeted With 'Swatting' Call	U.S. District Judge Tanya Chutkan appears to have been targeted by a "swatting" call. Police were falsely led to believe there was a shooting at her home. Officers quickly "determined no shooting took place," according to a police report.	U.S. District Judge Tanya Chutkan, who is overseeing former President Donald Trump's criminal election interference case in Washington, D.C., appears to have been targeted by a "swatting" call, in which police were falsely led to believe there was a shooting at her home. Officers quickly "determined no shooting took place," according to a police report obtained by HuffPost. Advertisement NBC News revealed on Monday that the home belongs to Chutkan. "Former Presidents enjoy no special conditions on their federal criminal liability," she wrote in her ruling last month. He called her a "true Trump hater," in all capital letters, in October and claimed that she would not be able to preside over a fair trial against him, according to ABC News. In 2021, more than 4,500 threats were made against U.S. judges as political tensions grew, Reuters reported. In 2022, a man was charged with attempting to kill Supreme Court Justice Brett Kavanaugh, CNN reported. Recently, authorities have been looking into threats against the Colorado Supreme Court judges who ruled that Trump's name could not be included on the state's primary ballot. As such, Chutkan, like many other judges, is no stranger to threats in the current political climate. Chutkan, who was nominated by Barack Obama, has developed a reputation for handing down sentences against Jan. 6 defendants that are more severe than what prosecutors had sought, according to The Associated Press. Advertisement Support HuffPost The Stakes Have Never Been Higher At HuffPost, we believe that everyone needs high-quality journalism, but we understand that not everyone can afford to pay for expensive news subscriptions. Our Life, Health and Shopping desks provide you with well-researched, expert-vetted information you need to live your best life, while HuffPost Personal, Voices and Opinion center real stories from real people. At HuffPost, we believe that a vibrant democracy is impossible without well-informed citizens. This is why we keep our journalism free for everyone, even as most other newsrooms have retreated behind expensive paywalls. Our newsroom continues to bring you hard-hitting investigations, well-researched analysis and timely takes on one of the most consequential elections in recent history.	2024-01-09 00:11:52	[0.011138527885328387, 0.02186892848835519, 0.007822372370187907, 0.10532377550326981, 0.0077128611284182115, 0.009060110539724575, 0.0053064600369092255, 0.8331754049361119, 0.5412613857906673, 0.015667675857192365, 0.002535833871693841, 0.0127485853095, 0.009350035934903102, 0.02065585538292073]		Judge-In-Trumps-Election-Interference-Case-Reportedly-Targeted-With-Swatting-Call.wav

News Embedding Model

The news embedding module is in charge of generating a real valued vector from a textual input. The goal of such a model is converting news Body text into its embedding in a space with N dimensions, with N equal to the number of categories inside the Training Dataset.

Dataset

The dataset was created by scraping news articles from Huffington Post . This dataset can be used to train models to classify news articles into different categories.

It Contains **6877 unique records**. The 14 categories and corresponding article counts are as follows:

- ARTS AND CULTURE: 1002
- BUSINESS: 501
- COMEDY: 380
- CRIME: 300
- EDUCATION: 490
- ENTERTAINMENT: 501
- ENVIRONMENT: 501
- MEDIA: 347
- POLITICS: 501
- RELIGION: 501
- SCIENCE: 350
- SPORTS: 501
- TECH: 501
- WOMEN: 501

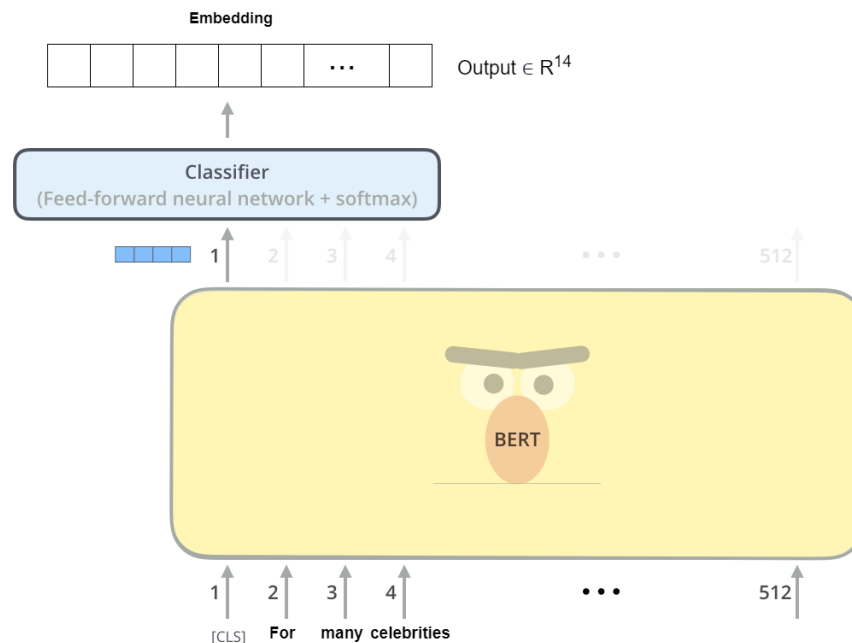
The dataset is available at the following URL link: [Dataset Link](#)

Model Architecture

In order to locate the news article in a 14 dimensional space, we decide to train a Classification model capable of predicting the news category, then remove the output layer

of the NN and use the Probability distribution vector as news embedding. In this way, as both news article and user are placed in a 14 dimensional space, we can compute the similarity between them with the cosine similarity of the two normalized vectors.

The model architecture is built upon the BERT Transformer model, specifically utilizing a streamlined version known as DistilBERT. The implementation of DistilBERT is provided by the Hugging Face Library. As shown in the illustration below we insert a FF Network that receives BERT output as input, and we use the output of the Softmax function as News Article Embedding.

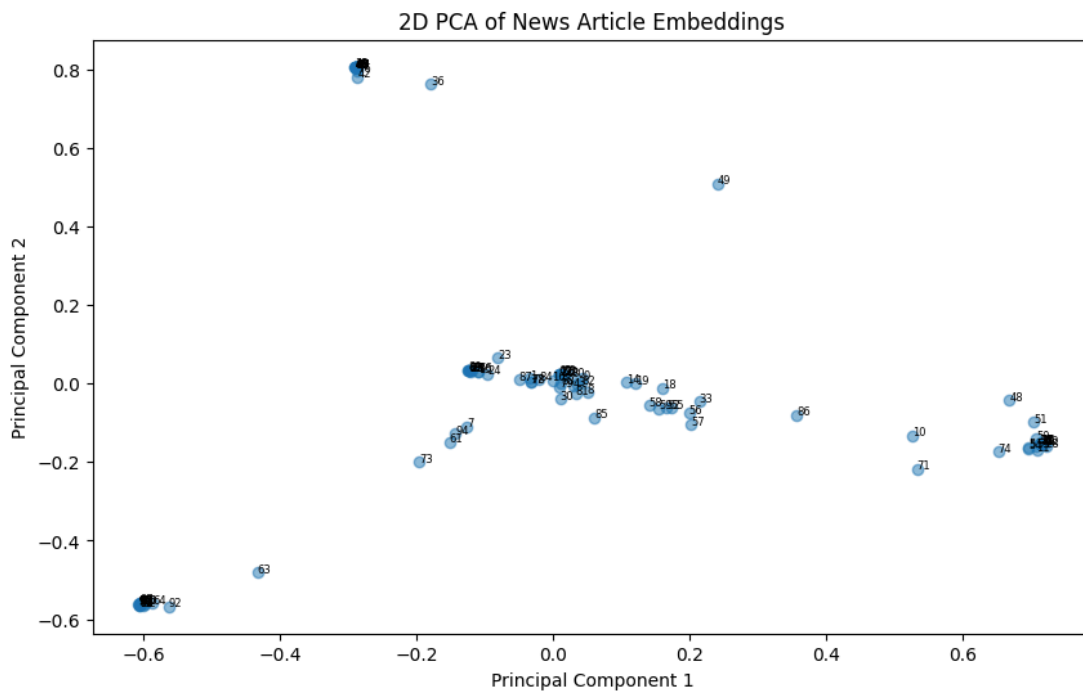


PCA of the collection

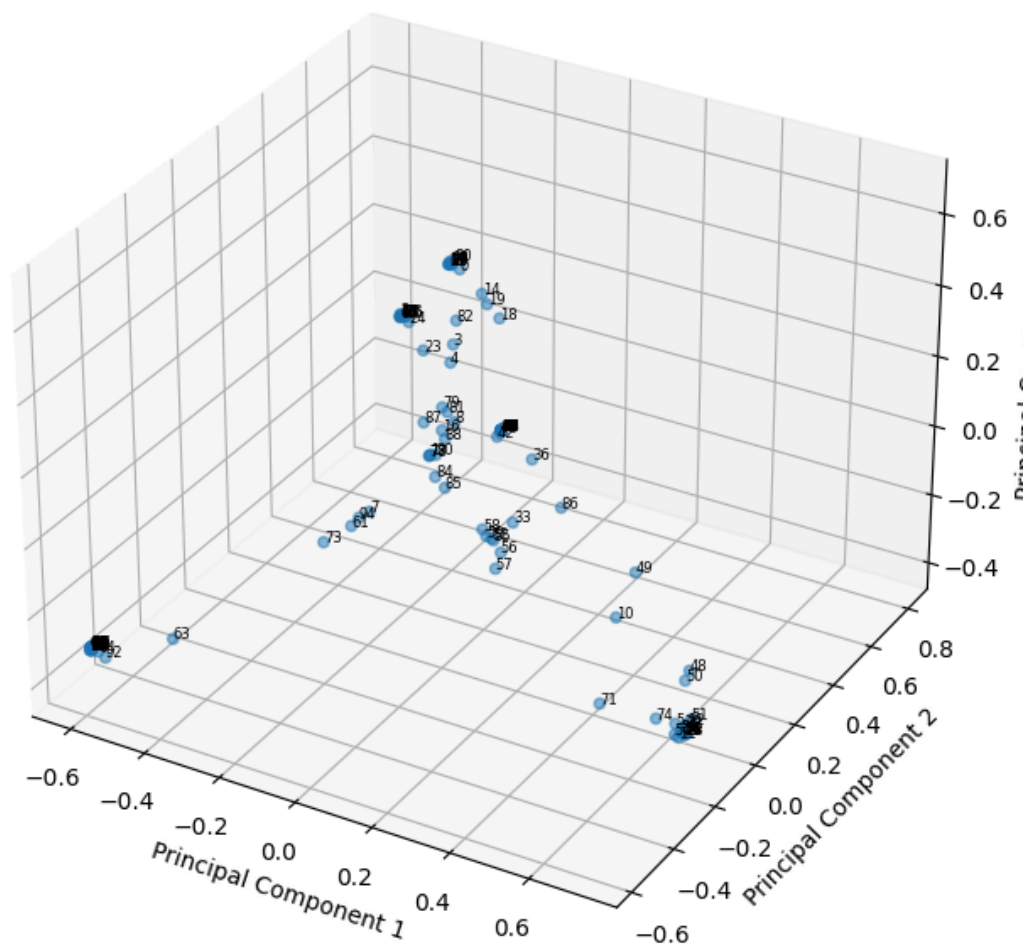
In this paragraph we insert the **Principal Component Analysis** (PCA) plot for the collection we obtained from the Indexer. We create a **2D** and **3D** PCA Plot to visualize the placement of news articles in the space and detect **clusters of articles**.

We obtained a final collection of news articles with following categories cardinality:

- ARTS AND CULTURE: 1
- BUSINESS: 19
- COMEDY: 3
- CRIME: 1
- EDUCATION: 1
- ENTERTAINMENT: 10
- ENVIRONMENT: 4
- MEDIA: 11
- POLITICS: 6
- RELIGION: 3
- SCIENCE: 16
- SPORTS: 16
- TECH: 6
- WOMEN: 3



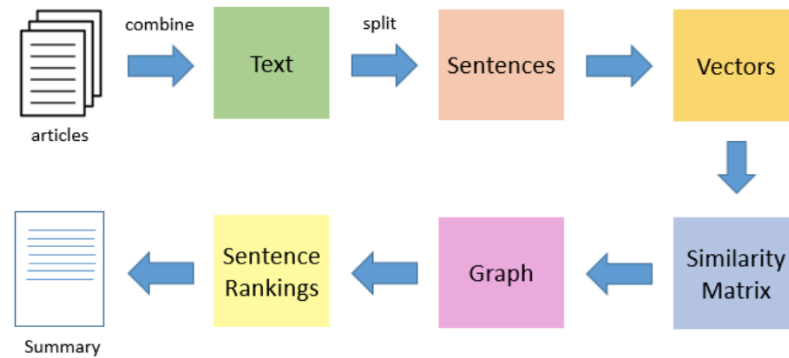
3D PCA of News Article Embeddings



Summarization Model

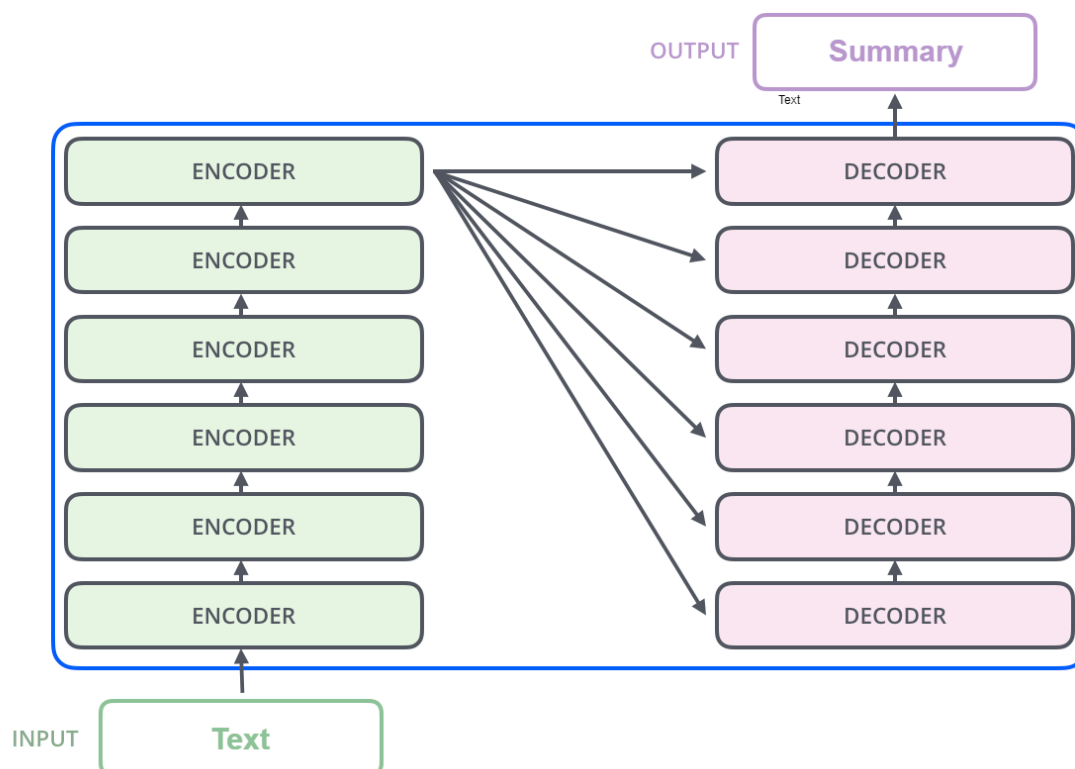
The summarization model is composed by two components:

1. **Latent Semantic Analysis (LSA) Summarize**, which extracts the most important sentences from a document and creates a shortened version of the news article. This step is required to feed a lower number of tokens into the input layer of the summarization model, because such number is limited.



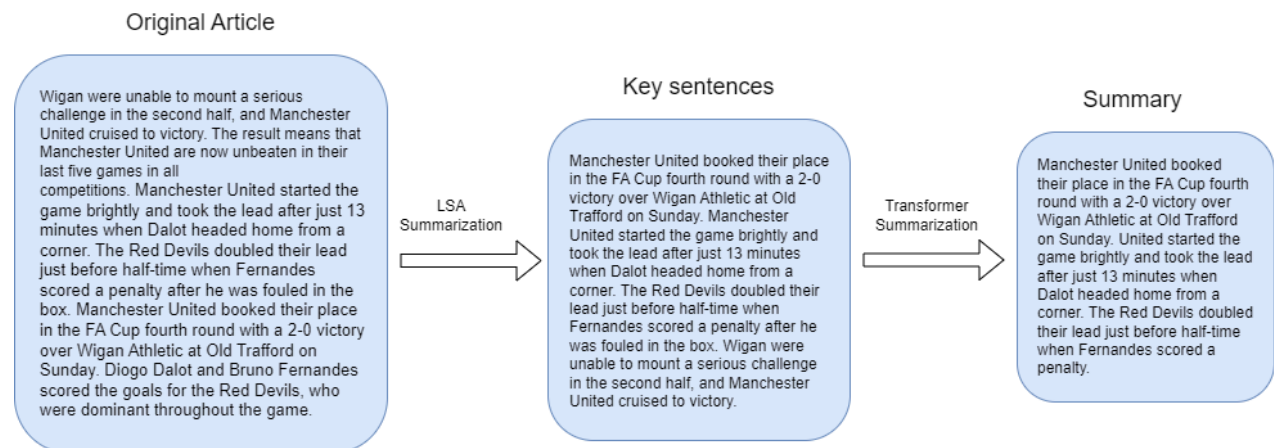
2. **Transformer based summarization**, which receive key phrases as input and create a short summary of the news article.

The interesting part of the Summarization model is the usage of BART Transformer architecture, which uses a standard seq2seq/machine translation architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT). In particular the Transformer has the following general architecture:



In particular we exploit the **BART large-sized model** introduced in the paper BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension by Lewis et al.

We propose in the following example the usage of the summarization model in the two steps, LSA Summarization and Transformer Summarization, on a random sport news:



Text to Speech Model

Coqui TTS [2] is a high-quality, open-source text-to-speech (TTS) engine that uses advanced neural network technology to generate natural-sounding speech.

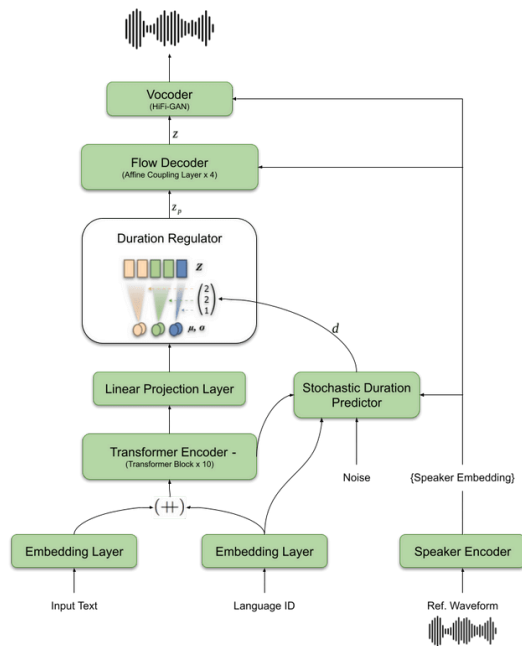
It can generate an audio track giving an input **text** and a **real speech sample**. We use this library to generate the news article audio given the article summary as textual input. The main idea is to provide a news player service similar to existent radio services where real anchormen announce the news.

We work following the list below:

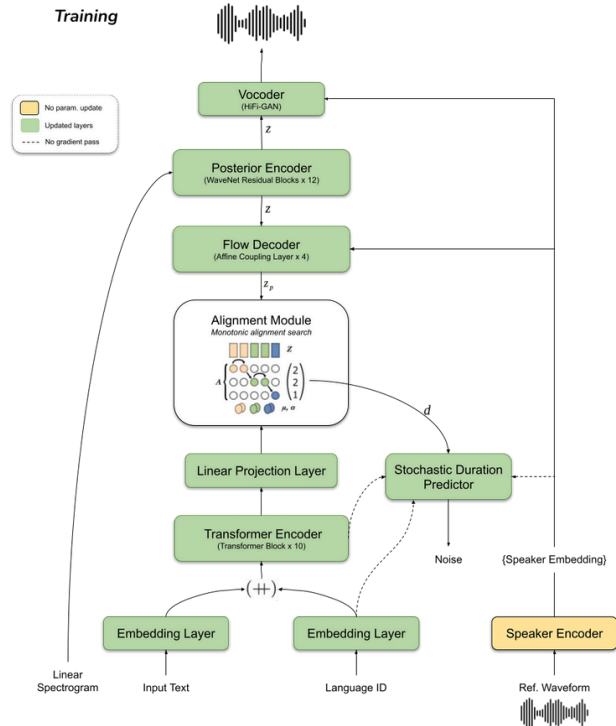
1. We **download a sample** of Obama voice as .mp3 file
2. The library creates a **TTS voice model** based on provided sample
3. We use the **TTS model** to create natural-sounding audio news

Coqui TTS use an architecture as the following, where the **Reference waveform** is the sample audio signal, **Language ID** is the specific language chosen for audio generation and **Input Text** is the text from which audio is created:

Inference



Training



Server

The server is implemented through a Flask web server that serves as the backend for a system with functionalities related to user registration, audio processing, speaker profiling, news suggestion, and user feedback.

Below is reported the list of the available endpoints:

Method	Endpoint	Description
GET	/	Renders the HTML page for user registration.
POST	/register	Registers a new user to the system along with audio and interests.
GET	/users	Returns the list of registered users.
GET	/speaker_profiles/<path:path>	Retrieves speaker profiles with the specified filename (username.pv).

GET	/audio-news/<path:path>	Retrieves audio news files with the specified filename.
POST	/feedback	Receives users' feedback regarding proposed news.
GET	/news_suggestion	Retrieves news suggestions based on given passengers' usernames.

Registration

Below is reported the registration page of TravelTales:

A user that wants to register a new account must follow this steps:

1. Enter **Username**: enter a desired username into the input field.
2. Specify **Interests**: find sliders corresponding to predefined interest categories (e.g., business, entertainment, politics, sport, tech) and adjust the sliders for each category to express your level of interest

3. **Start Recording:** click the "Start Recording" button to initiate the audio recording process, in order for the server to compute the user voice profile.
4. **Stop Recording:** once finished recording, click the "Stop Recording" button to end the audio capture.
5. **Review and Playback** (optional): optionally the audio track can be reviewed using the built-in audio player on the page. If satisfied with the recording, proceed to the next step.
6. **Submit Registration:** click the "Register" button to submit your registration details, including the recorded audio and interest levels, to the server.
7. **Wait for Confirmation:** wait for the server to process your registration. A success message will be displayed on the page if everything went fine.

Note: as the registration process includes the voice profiling, it should be carried out with the final devices that are also intended for the final use of TravelTales, in order to avoid bias introduced by different microphones devices.

Retrieval System

The server includes also the retrieval part, which consists in determining the collective interests of a group of users, scoring each news article based on how well it aligns with these interests, and then suggesting the top-ranked articles as recommendations for the users. Specifically, it receives as input the passengers representations and outputs the list of news which have maximum cosine similarity with the centroid of the passengers' representations. The scoring mechanism involves calculating the dot product between the embeddings, which is a mathematical operation used to quantify the similarity between vectors. In this context, it helps identify how closely the interests of users match those of each news article.

Client

The clients consists in several modules:

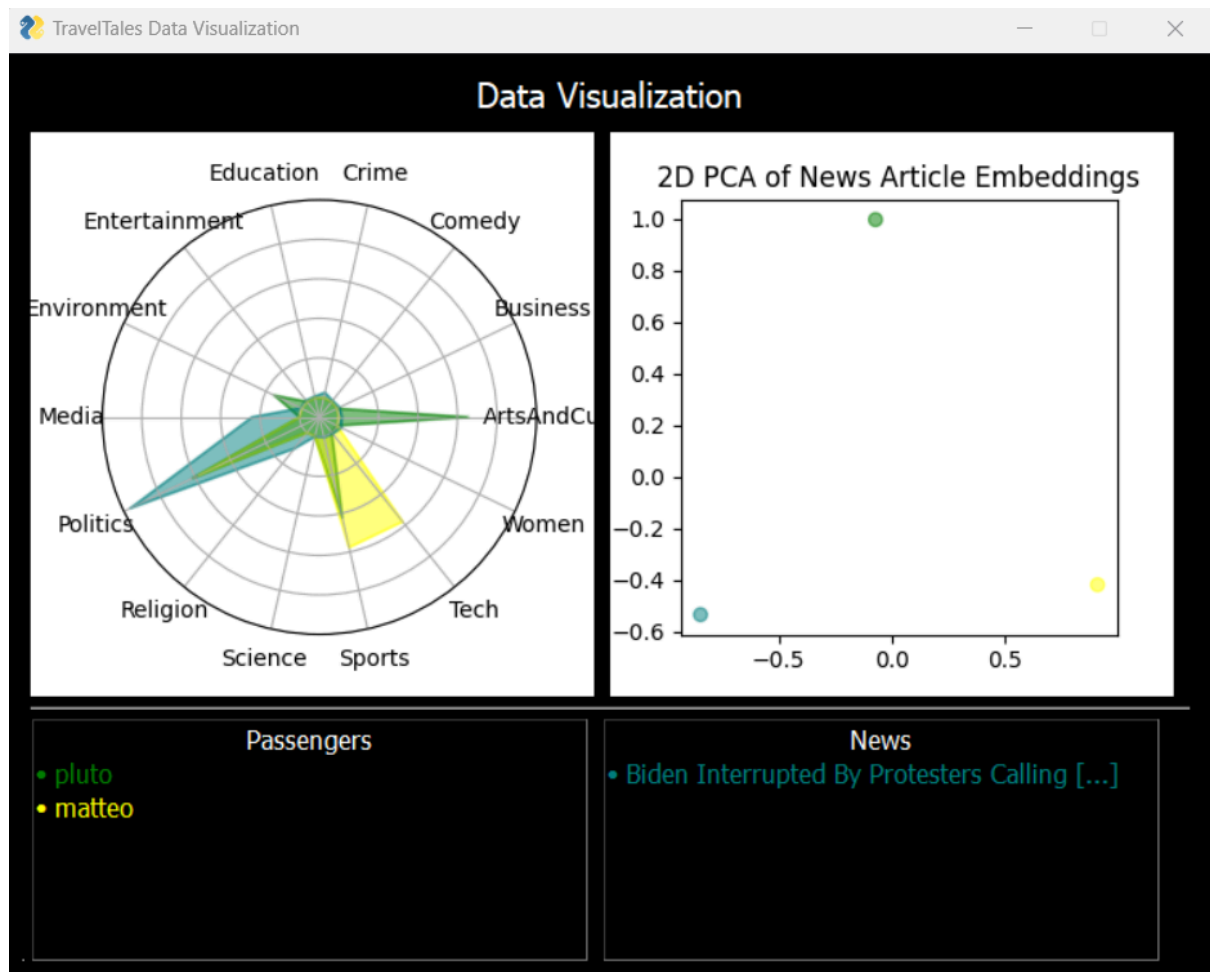
1. **News Retrieval & Player Module:** this module is in charge of managing the selection and playback of news articles in a system. Specifically, it oversees news suggestions, playback, and user interactions within a news-based system. It interacts with a server to fetch news suggestions based on the passengers' profiles. The goal is to provide a continuous stream of relevant news articles for the users to consume.
2. **Feedback Estimation Module:** this module is in charge of assessing user sentiment and thus engagement feedback for a given news article. To make it possible, it captures audio from passengers, processes it to gather insights, and sends the feedback to the server. It is divided into 2 submodules, respectively:
 - a. **Speaker Recognition Module:** it is in charge of implementing the real-time speaker recognition and allowing for the extraction and storage of speaker-specific speech segments while passengers are talking inside the car after a news has been proposed. It exploits the API from a library called *Picovoice* for real-time audio speaker recognition on constrained devices.
 - b. **Audio Sentiment Classification:** it is in charge of predicting sentiment from audio files, and estimating user engagement based on sentiment, speech

rate, and audio duration. The DNN used in this module is a pretrained Convolutional Neural Network that classifies the sentiment from an audio directly by extracting some audio-related features like MFCC (Mel-frequency cepstral coefficients).

Data Visualization Module

This Module is in charge of defining a GUI window for data visualization after a news has been proposed to a group of passengers. The window includes two plots:

- **Radar Diagram**
- **PCA Plot**



and displays information about users and news that lays in the same latent space. In particular, it is possible to visualize the correlation between the users' interests (specified at registration time) and the categories within which the news is associated. Again, in the 2-PCA plot we show the distance between the user and news embeddings in the latent space simplified in two dimensions, that is usually small.

Engagement Level Regressor

This module implements heuristics to derive the engagement to be associated with every passenger in the car that comments on the proposed news.

The Engagement level is a score between 1 and 10, and is estimated starting from 3 factors:

- Speech Rate
- Sentiment
- Audio duration

1. Speech Rate Calculation:

The computation of the speech rate consists in the ratio of non-silent frames to total frames in the audio using *librosa* library.

2. Audio Duration:

It calculates the total duration of the audio in seconds using the *pydub* library.

Sentiment to Engagement Mapping:

3. Sentiment of Audio:

The sentiment analysis is carried out on the total speeches of each passenger that commented on the proposed news. For each passenger speech we then have a sentiment associated.

→ Algorithm:

A mapping is then defined to map sentiment labels to engagement score:

Sentiment	Starting Engagement Score
neutral	2
disgust	2
calm	3
angry	5
fearful	5
sad	8
happy	9
surprised	10

Note: we associate a high starting score also to negative sentiment as long as our aim is to detect the generated engagement, thus if someone is feeling sad about a proposed news it does not necessarily mean that the engagement is low.

⇒ Adjustment based on Audio Duration:

The engagement score is adjusted based on different ranges of audio duration. Specifically:

Audio Duration Range	Engagement Score Adjustment
< 5 seconds	-1
10 - 15 seconds	+1
15 - 20 seconds	+2
>= 20 seconds	+3

⇒ Adjustment based on Speech Rate:

The engagement score is multiplied by the speech rate, and the result is rounded.

The final engagement score is constrained within the range [1, 10].

Speaker Recognition Module

The module contains a class called *SpeakerRecognizerStreaming* that plays a crucial role in facilitating real-time speaker recognition within the car environment. It is responsible for leveraging the [Picovoice Eagle API](#) [3], a specialized library designed for constrained devices, to seamlessly identify and differentiate speakers as passengers engage in conversations following the presentation of a news segment.

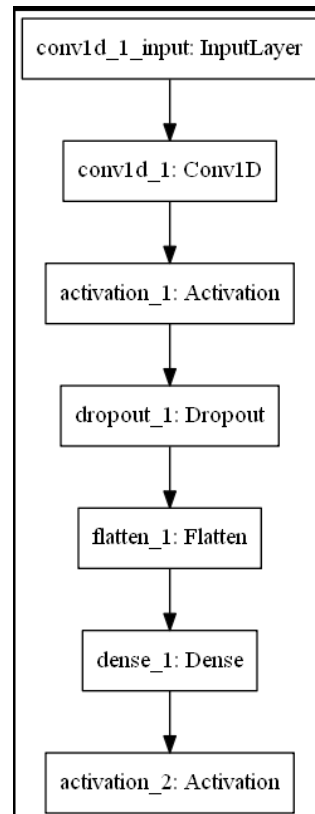
The implementation not only enables the instantaneous recognition of speakers but also facilitates the extraction and storage of speech segments specific to each individual. This functionality ensures that the system can effectively capture and analyze speaker-specific information during ongoing discussions inside the vehicle, providing valuable insights into passenger interactions and sentiments after the introduction of a news topic.

While a passenger is speaking and commenting on a proposed news, this module is responsible for extracting the speech (if greater than 4.0 seconds) and saving it on disk. Then, when silence is detected for at least 15.0 seconds, the process terminates and performs a merge of all the multiple speech segments from the same person (if any), and produces a single long .wav file for each one. Thus, these single files are used by the feedback estimation module to be fed into the audio sentiment classification and continue with the pipeline.

Audio Sentiment Classification

The module is implemented by a class called *AudioSentimentClassifier* that is responsible for sentiment prediction from audio files. This module employs a pre-trained Convolutional Neural Network (CNN), a deep learning architecture, adept at classifying sentiment directly from audio inputs. Leveraging the capabilities of the CNN, the classifier extracts essential audio-related features such as Mel-frequency cepstral coefficients (MFCC). These features serve as indicators for sentiment analysis.

The network consists in the following architecture:

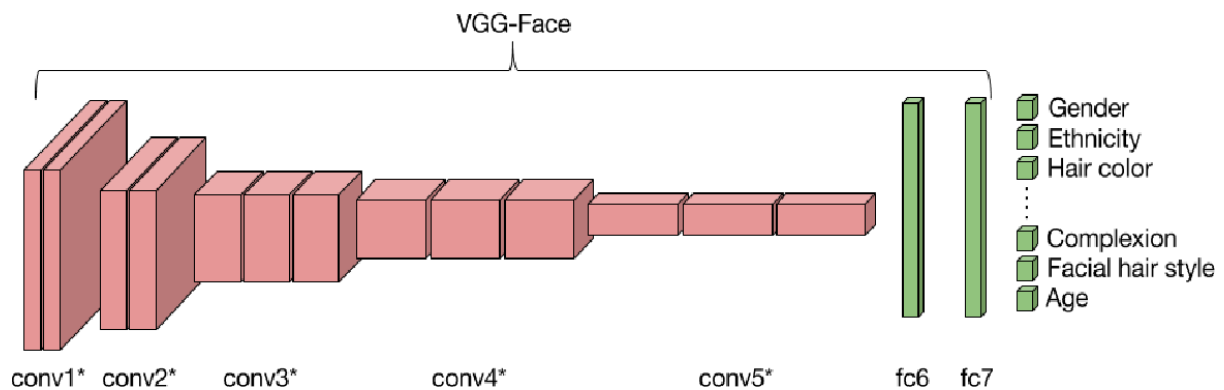


and was trained onto the [RAVDESS & TESS dataset](#).

Video Sentiment Classification

DeepFace [4], a Python library specializing in face recognition and attribute analysis, combines state-of-the-art models and efficient algorithms. This tool excels in facial verification, recognition, and detailed attribute analysis. We use DeepFace to extract emotional states in the Travel Tales user, to validate the sentiment prediction provided by the **Audio Sentiment Classifier**. The Video Sentiment Classification module can **detect multiple passengers** and can associate face to **passenger voice profile**.

We exploit the default configuration using VGG-Face model, which architecture is displayed below:



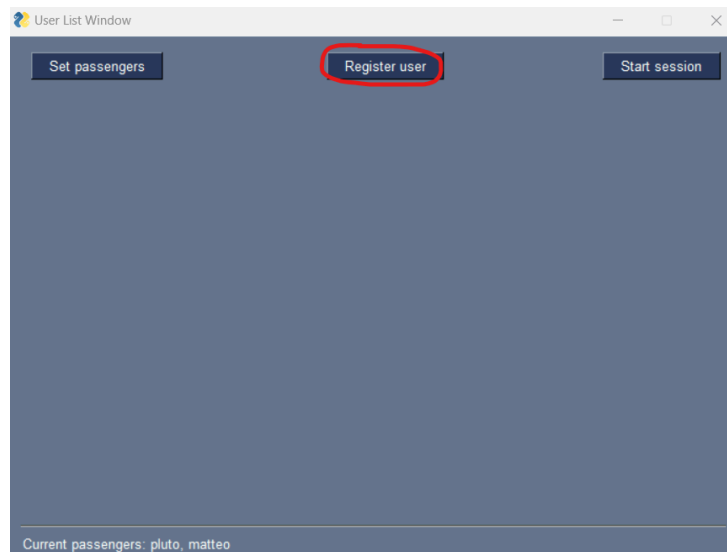
User Manual

A. User Registration

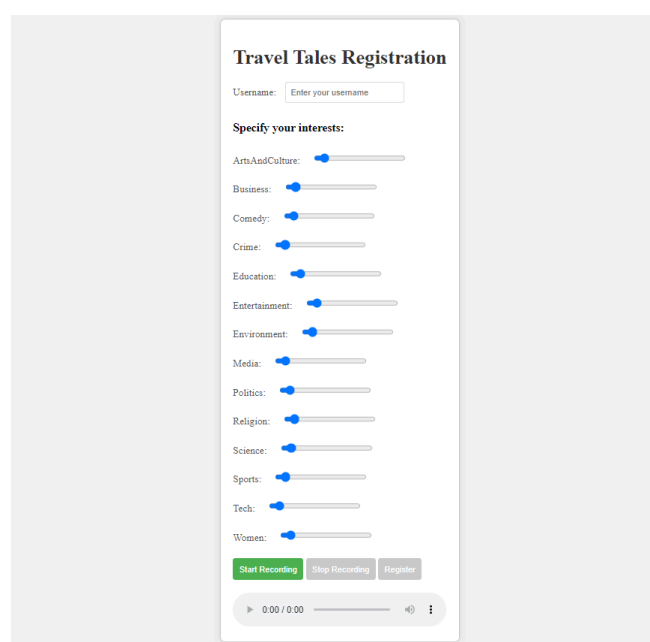
In order to register the user needs to click on the “Register user button” or to connect to the Server on port 5000:

localhost:5000

where a registration form is displayed.

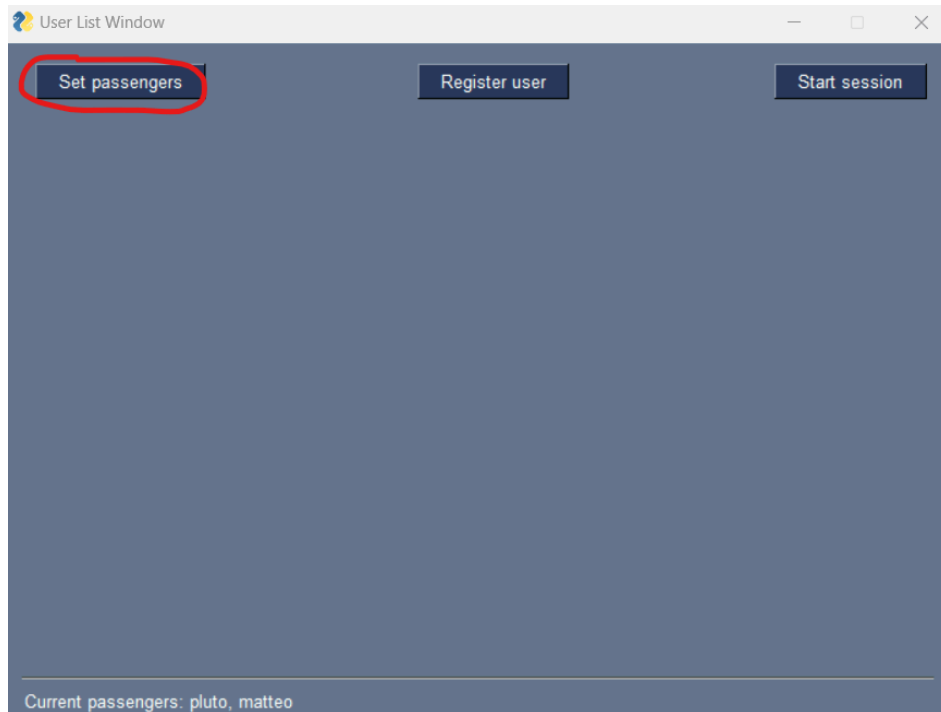


The user can insert its username, the interests on available categories and must register an audio sample reading the provided text to let the server elaborate the personal vocal profile. When all fields are filled, the user must click on the Register button and wait for the acknowledgement.

A screenshot of a registration form titled "Travel Tales Registration". The form is white and centered on a light gray background. It contains a "Username:" label followed by a text input field with the placeholder "Enter your username". Below this is a section titled "Specify your interests:" which lists 14 categories: ArtsAndCulture, Business, Comedy, Crime, Education, Entertainment, Environment, Media, Politics, Religion, Science, Sports, Tech, and Women. Each category has a horizontal slider with a blue dot indicating the selected level of interest. At the bottom of the form, there are three buttons: "Start Recording" (green), "Stop Recording" (gray), and "Register" (gray). Below these buttons is an audio player interface showing a play button, a progress bar with "0:00 / 0:00", a volume icon, and a settings icon.

B. On-board passengers selection

In order to select the passengers currently on board, the user must click on “Set Current Passengers” button, which allows the Client to send an HTTP request to retrieve the registered users list.



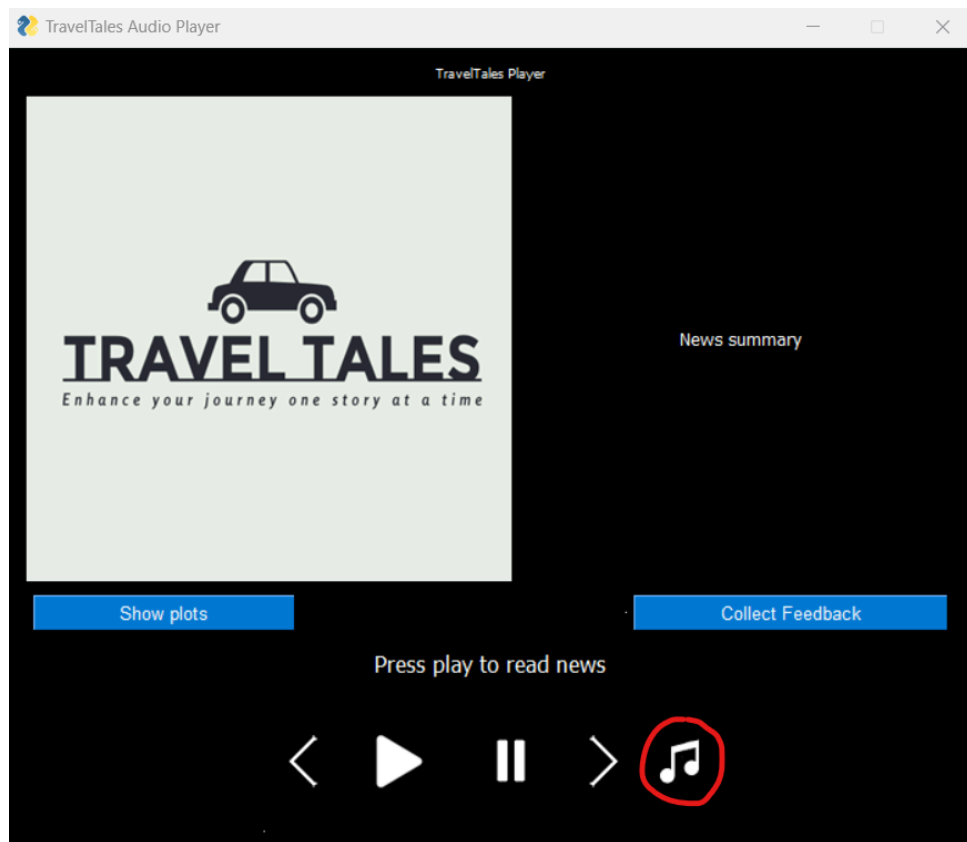
Then, a menu shows registered passengers and the user must specify which are the active passengers on board and then click the “Save” button.



After that, the user can click the “Start” button to execute Travel Tales startup.

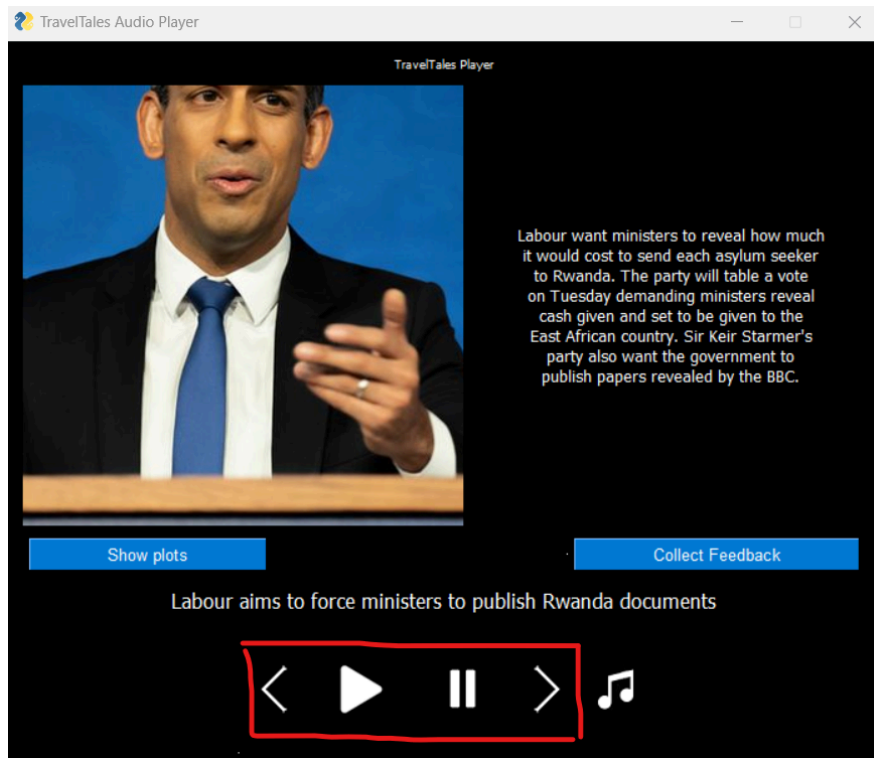
C. Music pause/unpause

When the application is running, a GUI is displayed to the user. The Music icon can be pressed to pause/unpause the ambient music.



D. News player commands

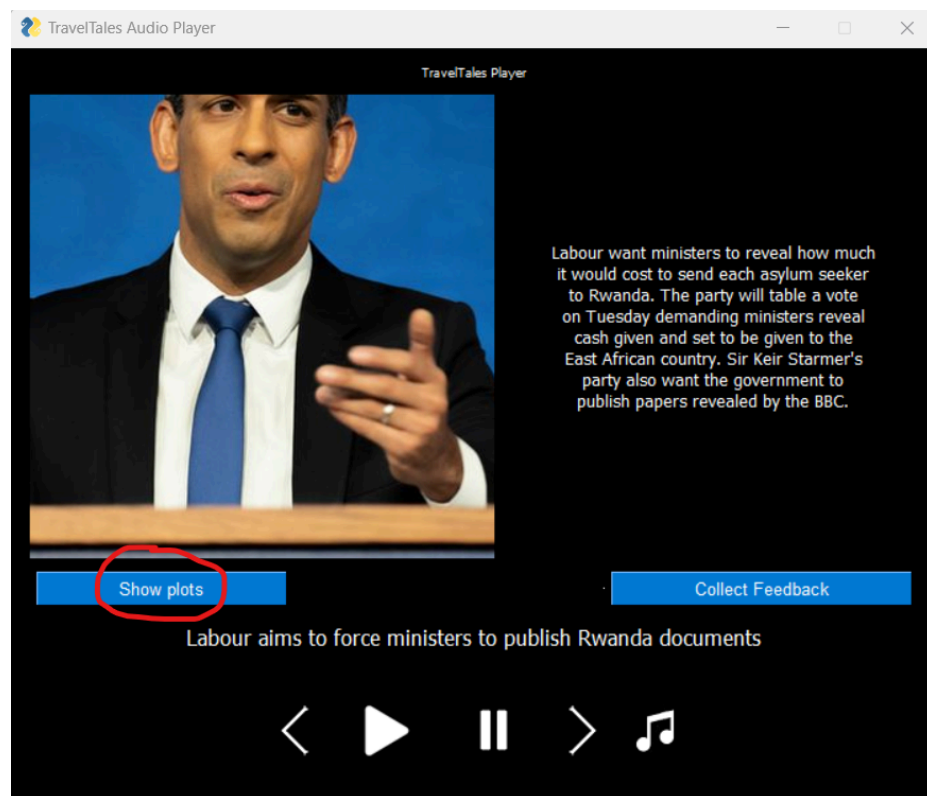
The news player panel allows the user to play, pause and switch suggested audio news. When the “Pause” button is pressed, the ambient music restarts.



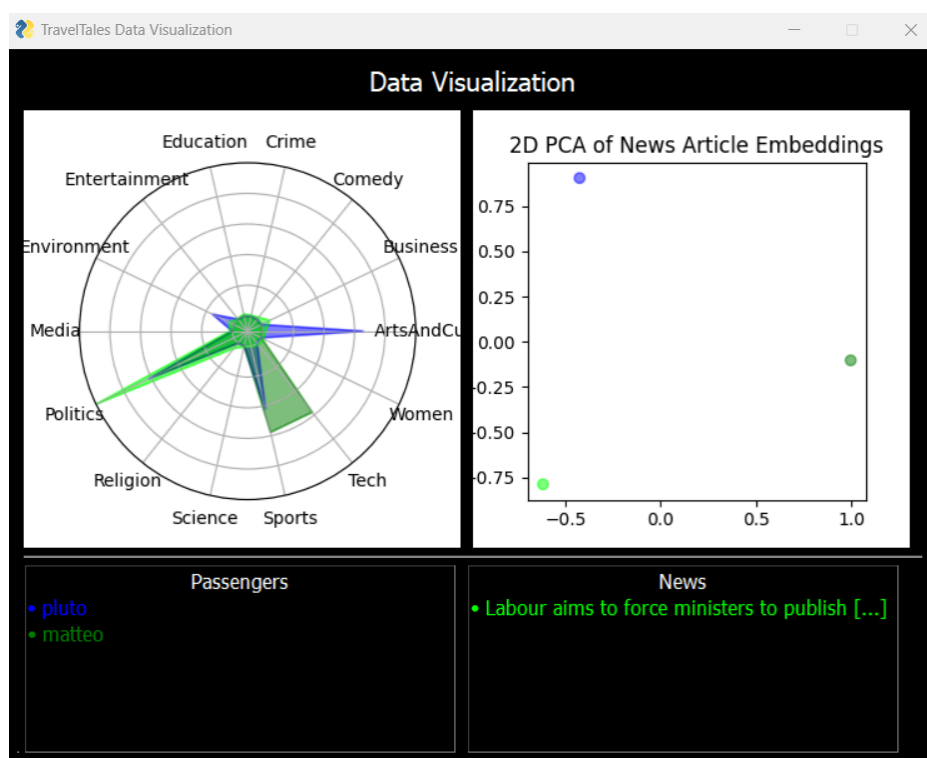
Above the news player panel, there are the news title, summary and thumbnail.

E. Plots panels

Clicking the “Show plots” button, the Plots Panels opens.

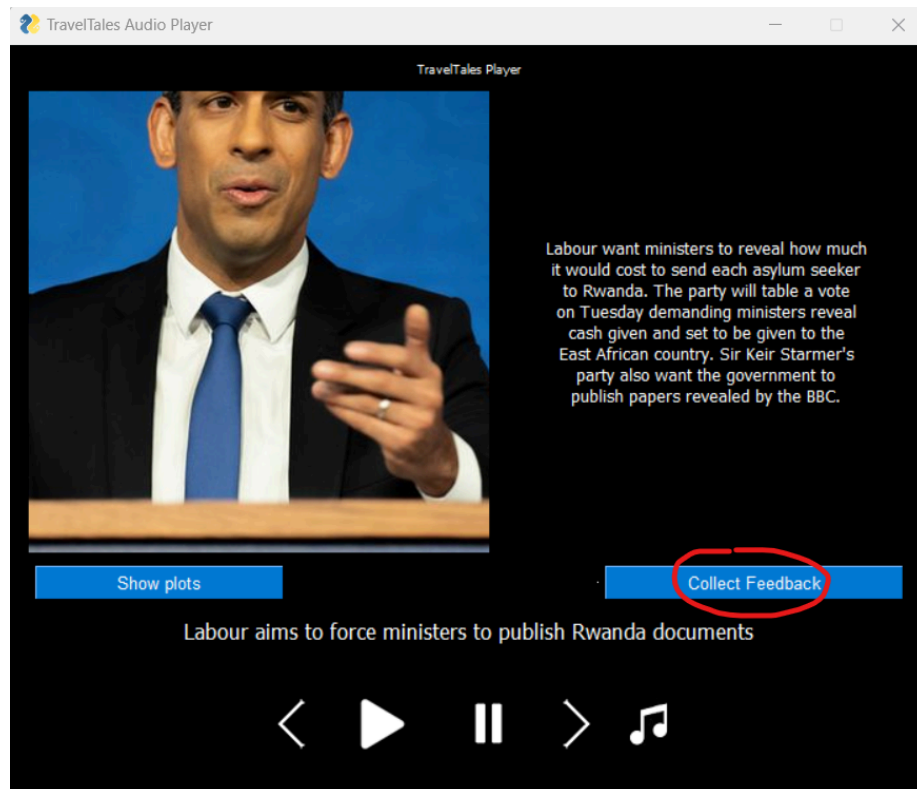


The panel shows both PCA and Radar chart plots with a colorized legend of news and passengers.

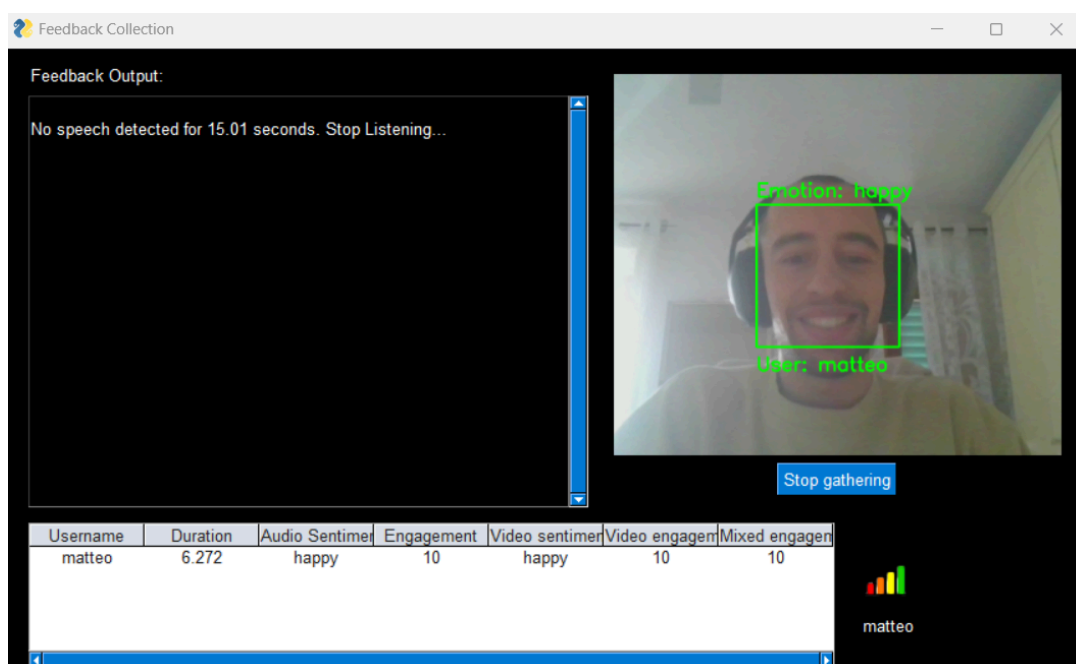


F. Feedback Gathering (with camera)

Clicking the “Collect feedback” button, the Feedback Gathering Panel opens.



In the panel a confidence score of the audio signal and a camera view with detected sentiment is displayed. After a silence period is detected, a Table with Username, Audio signal duration, Audio sentiment, Video sentiment and Engagement level are shown. The “Stop gathering” button can be clicked to immediately stop the feedback gathering operations.



G. Explicit feedback form

In order to obtain explicit feedback from the application users, the GUI provides a table with clickable entries as shown in the image below.

Feedback Output:

No speech detected for 15.01 seconds. Stop Listening...

Stop gathering

Username	Duration	Audio Sentiment	Engagement	Video sentiment	Video engagement	Mixed engagement
matteo	20.256	angry	3.95	NA	3.95	3.95

matteo

Once the entry is clicked the user is taken to a browser page with a form to insert the explicit feedback.

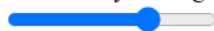
Collecting feedback for matteo

[News link](#)

Select the emotion that the news generated to you:

happy ▼

What was your engagement level?



Submit

The explicit feedback can be exploited to create a dataset useful to train an engagement estimation model.

Conclusions

In conclusion, the TravelTales in-car conversation recommendation system offers a notable advancement in addressing the dual challenges of enhancing in-car entertainment while minimizing driver distractions. Recognizing the dynamic nature of user preferences, a potential avenue for future improvement lies in the development of a server-based system that could utilize passenger feedback gathered during the recommendation process to iteratively update user embeddings, thereby refining the accuracy and relevance of personalized conversation prompts. Such an approach ensures the continual adaptability and optimization of the in-car conversation experience.

It is important to acknowledge the challenges associated with obtaining a suitable dataset for training an updating model, particularly given the problems of tuning the crucial parameter of step size in the update process crucial for the successful implementation of the update function.

References

- [1] “A multimodal approach for modeling engagement in conversation” by F. Morreale, C. Cosi, L. Benech, E. Nisi, and V. Pallotta, published in Frontiers in Computational Neuroscience, 2023, DOI: 10.3389/fcomp.2023.1062342.
- [2] Coqui TTS Library by Coqui AI, released under the MIT License and currently in version 0.22.0. The project's comprehensive documentation is available at <https://docs.coqui.ai/>: <https://docs.coqui.ai/>.
- [3] Picovoice by Picovoice Inc., released under the Apache License, Version 2.0 and currently in version 2023.12.06. The project's comprehensive documentation is available at <https://picovoice.ai/docs/>: <https://picovoice.ai/docs/> and is also available on GitHub at <https://github.com/Picovoice/picovoice>.
- [4] DeepFace Library by the DeepFace Lab at the University of California, Berkeley, released under the MIT License and currently in version 0.0.55. The project's comprehensive documentation is available at <https://github.com/serengil/deepface>