DEPARTMENT OF COMPUTER, MODELING, ELECTRONIC
AND SYSTEMS ENGINEERING

Master's Degree Couse in Computer Engineering

"IoT Sensors and Device Programming" course project report

Authors
Angelica Porco (S. N.)
Francesco Carmelo Capria (S. N. 263664)
Giuseppe Mattia Greco (S. N. 264526)
Hanna Andrieieva (S. N.)

Supervisor
Prof. Giancarlo Fortino

Introduction

A Smart Museum represents a museum environment enhanced by digital and intelligent technologies to improve the interaction between visitors and the exhibited artworks. Unlike traditional museums, it integrates *IoT sensors* and *data analysis techniques* to collect information on visitors' behaviors and emotions. This real-time data provides museum managers with advanced tools to optimize the visitor experience and improve the management of exhibition spaces.

One of the key aspects of Smart Museums is their ability to analyze visitor sentiment, meaning their opinions on the exhibited artworks and the museum's layout. Traditionally, sentiment analysis relies on explicit feedback such as questionnaires or written reviews, which are often not very spontaneous and limited in scope. However, the adoption of IoT technologies combined with ***Natural Language Processing*** (**NLP**) techniques enables the collection of implicit feedback based on naturally acquired audio recordings, overcoming the limitations of traditional methods. This project aims to develop an innovative sentiment analysis system within a Smart Museum by using IoT devices to capture and process voice data.

Main objectives

- **Analysis of optimal microphone placement**
  Study the optimal distribution of audio capture devices to maximize recording quality, reduce environmental noise, and ensure targeted data collection near the exhibited artworks.

- **Development of a Java solution for PC**
  Implement a sentiment analysis system based on *NLP techniques*, leveraging devices with high computational power for data processing.

- **Creation of a Python version using an Arduino board**
  Develop a lightweight sentiment analysis model capable of processing voice data using IoT audio collection devices.

- **Data analysis**
  Create a real-time dashboard using Node-RED to visually represent and interpret the obtained results.

The implementation of this system will enable a more accurate and timely analysis of visitors' opinions, providing museum managers with tools to improve the overall experience. Additionally, the study of microphone placement and listening stations will help refine the quality of implicit feedback, ensuring more reliable and detailed analysis.

Study of microphone placement

The placement of audio capture devices, such as Arduino and PC, within the Smart Museum is a crucial element to ensure effective and targeted data collection, allowing the sentiment of visitors to be gathered in a precise and contextualized manner. Each device must be strategically positioned to optimize recording quality and collect specific or general feedback depending on the area of the museum.

PC station placement

The PC station is designed to collect and analyze aggregated data from the Arduinos and to display real-time information on visitor sentiment. The placement of the PC is crucial for enabling continuous interaction between visitors and the collected data, with real-time visualization of the results.

The PC station is located in **reflection areas** of the museum, such as couches and tables, where visitors take a break to reflect on the museum experience. These areas are ideal for gathering general feedback on the overall experience rather than on individual artworks. While the Arduinos collect specific data on the artworks, the PC station provides an aggregated overview and displays both the sentiment of individual pieces and the overall museum experience.

In the reflection areas, visitors can see a real-time dashboard that graphically represents the sentiment gathered from conversations. The dashboard provides detailed information on how visitors perceive individual artworks, through the data collected by the Arduinos, as well as the overall museum experience. This real-time visualization allows visitors to receive immediate feedback on their emotions and opinions, creating a direct connection with the museum and its artworks. The presence of the PC station, therefore, not only collects and analyzes data but also serves as an interactive tool for the audience, stimulating active engagement.

An example of device placement is as follows:

Museum microphone placement example

Arduino placement

Arduinos are lightweight devices, ideal for being placed near artworks. Their placement near the most important works, whether paintings or sculptures, allows for targeted and direct feedback on each piece, enabling the analysis of visitors' reactions and emotions in real-time. Arduinos can be positioned just a few centimeters away from the artwork at a height of approximately 1.5 meters, which corresponds to the average height of a visitor. This height optimizes recording quality by capturing natural and spontaneous conversations without interference from ambient noise or other sources.

PC version

The system developed for the PC version in **Java** is designed to collect and analyze voice feedback from museum visitors. The application, which uses a *speech recognition engine* and a *sentiment analysis library*, allows the visitors' speech to be captured and transformed into text while analyzing the emotions expressed. Based on the language of the text, whether Italian or English, the sentiment analysis classifies emotions as *positive*, *negative* or *neutral*. The results are displayed in real-time through an interactive dashboard powered by Node-RED, providing the museum staff with an overview of visitors' reactions to the artworks and insights into the overall museum experience.

Tools and tecnologies used

- **Java**
  The primary programming language is used for the development of the system, particularly for speech recognition and sentiment analysis.

- **Vosk**
  A speech recognition and speech-to-text library, used to transcribe audio in real-time. It supports both Italian, English and a few more languages.

- **Stanford CoreNLP**
  Stanford CoreNLP is a Java-based library that processes English text to detect grammar, meaning, and sentiment. It helps analyze and understand natural language.

- **Hugging Face model**
  By using REST API, we used the *tabularisai/multilingual-sentiment-analysis* to provide fast sentiment analysis for the captured phrases.

- **Node-RED**
  Node-RED is an open-source, flow-based development tool for wiring together hardware devices, APIs, and online services. It provides a visual interface where users can create workflows by connecting nodes, which represent various devices or functions, such as sensors, databases, or messaging protocols like MQTT.

- **MQTT protocol**
  MQTT (Message Queuing Telemetry Transport) is a lightweight, open-source messaging protocol designed for low-bandwidth, high-latency, or unreliable networks. It uses a publish/-subscribe model where clients send (publish) messages to a broker, and other clients (subscribers) receive messages based on topics they are subscribed to.

- **Grafana**
  A tool used for the graphical visualization of the analyzed data, creating dashboards to monitor sentiment in real-time.

- **InfluxDB**
  A time-series database used to store data related to sentiment and transcribed text, allowing for historical analysis.

Development

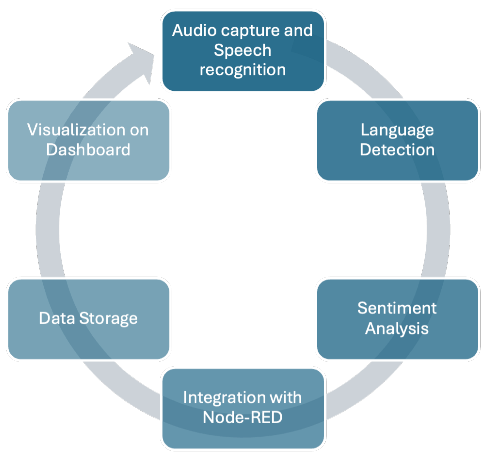
The system development process was divided into several phases. The first phase involved implementing the speech recognition engine, integrating Vosk to transcribe speech in real-time. At the beginning of each session, the user is prompted to select a preferred language—either Italian or English. Based on this choice, the application dynamically loads the corresponding Vosk model for speech recognition. Audio is captured from the PC's microphone through the `AudioCapture` class, which continuously listens and streams audio data. This stream is passed to the `SpeechToText` component, where it is transcribed using the selected Vosk model.

Once the spoken language is converted to text, the system proceeds with sentiment analysis. The next step depends on the language selected:

- If Italian was selected, the system sends the transcribed text to a cloud-based model *hosted* on ***Hugging Face*** (*tabularisai/multilingual-sentiment-analysis*). This model is trained exclusively on synthetic multilingual data generated by advanced LLMs, ensuring robust performance across different languages and cultural contexts. The interaction is handled by the `SentimentAnalysis` class, which sends a REST request to the Hugging Face API and receives a sentiment classification—positive, negative, or neutral—along with the detected language.

- If English was selected, instead of using the Hugging Face API, the system activates a local instance of ***Stanford CoreNLP***, a powerful Java-based NLP toolkit. CoreNLP performs advanced linguistic analysis on the transcribed English text, leveraging syntactic and semantic structures to determine the sentiment. The analysis is performed locally within the Java application to ensure low-latency and accurate classification.

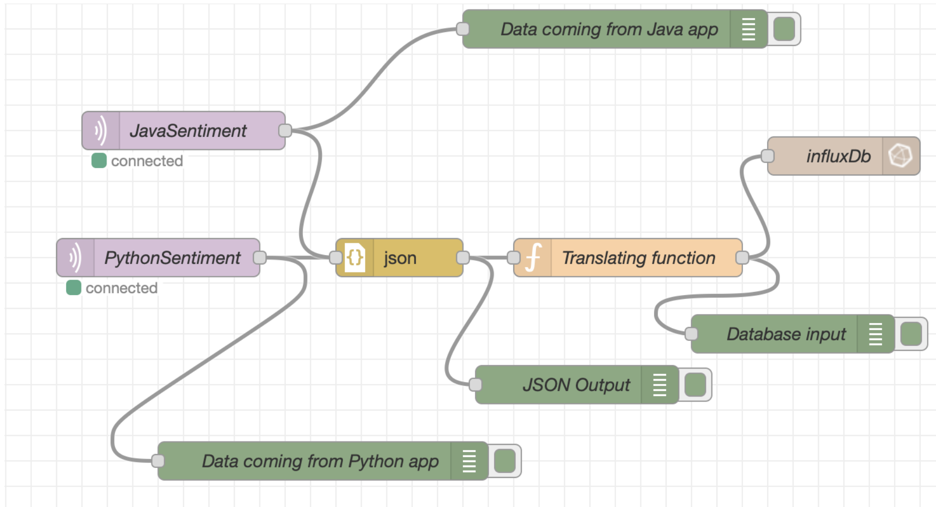
This language-aware architecture ensures that both transcription and sentiment analysis are tailored to the selected language, enhancing accuracy and adaptability in multilingual environments. This classification is then packaged together with the original text and forwarded to the `MQTTClient` module, which manages the communication with the MQTT broker. The data is published on a specific topic and made available to external services.



Development chart

The workflow managed by Node-RED handles receiving data from the Java application. This data is sent using MQTT protocol to a Node-RED node. The data, which includes the analyzed text, sentiment, and language, is processed by the node, which verifies the presence of all necessary data in the payload and converts it into the correct format for insertion into the Influx database.

The sentiment data is then stored in InfluxDB, a time-series database, to collect real-time information about the visitors' sentiment. The information is displayed on a Grafana dashboard, directly fed by Node-RED, allowing observation of the evolution of emotions expressed by visitors based on the artworks and the overall museum experience.



Node-RED flow

Arduino version

As with the Java version, the version developed in **Python** has as its focus the need to record feedback coming from a public place (e.g., a museum) through a microphone and perform sentiment analysis on the collected feedback.
The major difference is that the device used as a microphone is an **Arduino Portenta H7** with the **Vision Shield** included: an addon board providing machine vision capabilities and additional connectivity to the Portenta family of Arduino boards, designed to meet the needs of industrial automations.

A script was written for the Arduino that captures the audio from the microphone on the Vision Shield; the audio is then processed by the PC to which the board is connected.

Tools and technologies used

- **Python**
  A high-level, general-purpose programming language.

- **Arduino Portenta H7**
  Arduino is a hardware and software platform consisting of a series of electronic boards equipped with a microcontroller and accompanying software that allows for simplified programming.

- **Vosk**
  A speech recognition library, speech-to-text, used to transcribe audio in real-time. It supports both Italian and English.

- **Transformers (Python library)**
  Transformers provides APIs to easily download and train pre-trained models. By using the pipeline module, we were able to use Hugging Face models in an easy way.

- **MQTT protocol**
  Is a lightweight publish–subscribe, machine-to-machine network protocol for message queue/message queuing service.

Development

This implementation follows the same architecture and objectives as the Java version, aiming to achieve equivalent functionality. Our challenge was to use the Arduino board like an audio source (a microphone) by coding on it a script that records all the audio.

The script is written in C++ language ad uses the PDM library that allows the use of Pulse-Density Modulation microphones such as the one found on the Vision Shield. In particular, the script initializes the microphone and, using a buffer to store the captured audio. This buffer is then read by the Python script running on the PC to perform the speech-to-text. On the computer side, the `audio_capture` module listens to the serial port and reads the incoming data in fixed-length frames. It ensures **synchronized** and **uninterrupted** streaming of the audio by handling buffering and error control. Once captured, the audio is passed to the `speech_to_text` module, which utilizes the Vosk speech recognition engine. The model loaded is optimized for multilingual support, particularly Italian and English, and is capable of performing real-time transcription even on consumer-grade hardware.

Arduino version chart

After converting speech into text, the `sentiment_analysis` module performs sentiment classification. Unlike the Java version — which sends data to a Hugging Face API — the Python version uses a *locally downloaded* version of the **tabularisai/multilingual-sentiment-analysis** model from Hugging Face. This model is run through the transformers library with a `pipeline("sentiment-analysis")`, enabling low-latency, offline predictions. It is specifically trained on synthetic multilingual datasets generated by large language models (LLMs), which ensures robustness across various languages and cultural expressions. The model returns both the sentiment label (positive, negative, or neutral) and the detected language of the input.

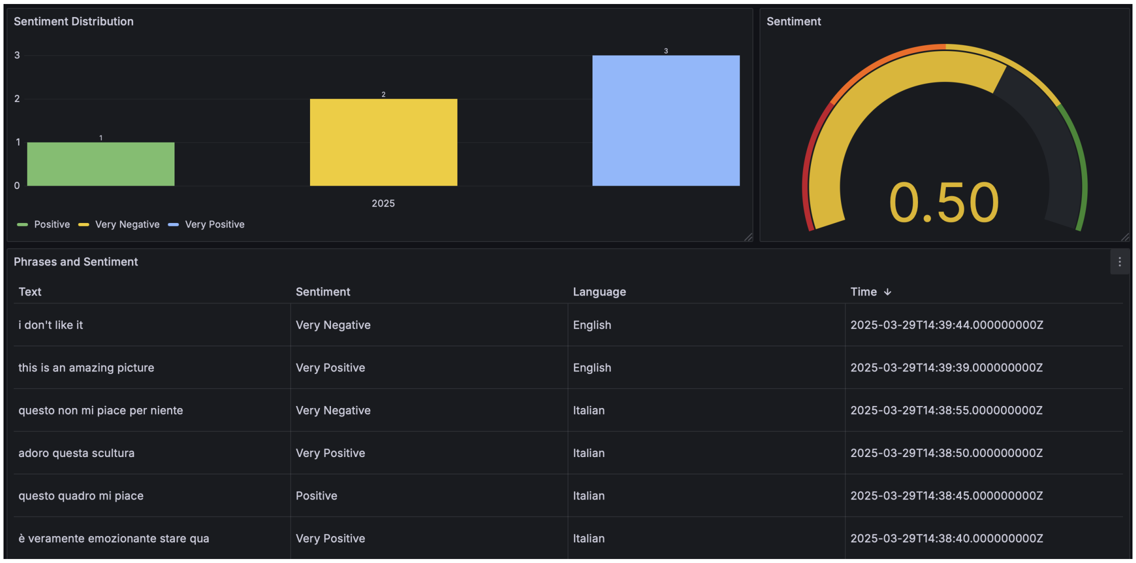
The processed data, including the transcribed text, sentiment, and language, is then published using the MQTT protocol through the built-in client in the application. Messages are sent to a Mosquitto broker, where they are received by Node-RED, formatted, and inserted into an InfluxDB instance for time-series storage. This data is then visualized in Grafana, which provides dynamic dashboards showing sentiment trends over time.

Results

The system successfully demonstrated its ability to acquire audio input, transcribe it into text, and perform sentiment classification in real time. Functional tests confirmed the proper interaction between all components, including speech recognition (via Vosk), sentiment analysis (using

Hugging Face and Stanford CoreNLP), and message exchange through the MQTT protocol.

Audio data captured by the devices was correctly transcribed in both Italian and English, and the resulting text was classified into positive, neutral, or negative sentiment. The data was then visualized through a real-time dashboard, as shown below.



Grafana dashboard

The real-time dashboard, developed with Grafana and fed by data processed through Node-RED and InfluxDB, provides a clear and dynamic visualization of sentiment trends. As shown in figure, the interface presents a time series graph where each line represents the intensity of a specific sentiment (positive, neutral, or negative) over time.

This visualization allows users and curators to observe fluctuations in visitor emotions and analyze how sentiment evolves throughout the day or in response to specific events or artworks. The clean layout and color-coded lines help in quickly identifying emotional peaks or general trends, supporting future decision-making for exhibition design or visitor engagement strategies.

Future development

In the future, the system could be updated to handle multiple languages simultaneously, enabling parallel management of sentiment data in Italian, English, and other languages.

In terms of hardware evolution, the Arduino-based version could be upgraded to support on-board processing by deploying optimized machine learning models with TensorFlow Lite Micro. This would reduce reliance on the PC and enable fully autonomous sentiment analysis on edge devices, improving scalability and reducing latency. Similarly, the PC version could benefit from GPU acceleration or hybrid CPU-GPU inference to handle larger volumes of voice data and support real-time multi-user interaction.

From a software perspective, the system could be enhanced with context-aware sentiment analysis, integrating metadata such as time of day, artwork proximity, or visitor behavior (e.g., time spent in front of an exhibit). This would enable a richer interpretation of emotions and allow museum curators to tailor the visitor experience accordingly.

Finally, in terms of usability and deployment, the system could evolve into a plug-and-play framework for smart environments beyond museums—such as retail spaces, events, or educational institutions—where understanding public sentiment is valuable. Open-sourcing the project or packaging it as a modular platform with simple configuration options would greatly facilitate adoption by third parties.