Pontifícia Universidade Católica
do Rio de Janeiro

José Matheus Carvalho Boaro

An Architecture for Enhancing Real-Time Multimedia Flows with Semantic Information

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Prof. Sérgio Colcher

Rio de Janeiro
September 2023

Pontifícia Universidade Católica
do Rio de Janeiro

José Matheus Carvalho Boaro

An Architecture for Enhancing Real-Time Multimedia Flows with Semantic Information

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Prof. Sérgio Colcher
Advisor
Departamento de Informática – PUC-Rio

Prof. José Alberto Rodrigues Pereira Sardinha
Departamento de Informática - PUC-Rio

Prof. Júlio César Duarte
Instituto Militar de Engenharia - IME

Profª. Débora Christina Muchaluat Saade
Universidade Federal Fluminense - UFF

Rio de Janeiro, September 4th, 2023

José Matheus Carvalho Boaro

Graduated in computer science by the Federal University of Maranhão.

Bibliographic data

Boaro, José Matheus Carvalho

An Architecture for Enhancing Real-Time Multimedia Flows with Semantic Information / José Matheus Carvalho Boaro; advisor: Sérgio Colcher. – 2023.

64 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2023.

Inclui bibliografia

1. Informática – Teses. 2. Arquitetura. 3. Multimídia. 4. Robôs Móveis. I. Colcher,Sérgio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Acknowledgments

To my family, specifically my mother and sister, for all the support given to me during this journey, through their support I was able to go further.

To Louise França, who has been by my side since the beginning of my academic life, during the moments of difficulty, doubt and fear, knowing that you were by my side was fuel for me to overcome them.

To my journey friends, you were all essential during this path. With you I was able to share the victories and also the defeats, always being able to count on your unconditional support.

To my laboratory colleagues, who were essential for me to get here. Your unconditional help enabled me to better understand several topics that were unknown and hostile to me, as well as emotional support in times of doubt.

This study was partially financed by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) - Finance Code 131661/2021-3, and by the Air Force Office of Scientific Research under award number FA9550-22-1-0475.
This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001

Abstract

Boaro, José Matheus Carvalho; Colcher,Sérgio (Advisor). **An Architecture for Enhancing Real-Time Multimedia Flows with Semantic Information**. Rio de Janeiro, 2023. 64p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

While traditional multimedia systems focused on efficient coding and storage of media types and their temporal relationships, the current demand for rich and customized experiences calls for a deeper understanding of semantic content. In this study, we propose the integration of semantic-level processing into multimedia systems, enriching content with information about real-world entities, such as objects, actions, agents, and language interpretation. The main contribution of this dissertation is the proposal of an architecture for real-time multimedia data enhancement that is able to use machine learning techniques to extract semantic representations and incorporating it into multimedia data streams as a native and basic service. To provide a concrete demonstration of the proposal, we implement two use cases that serve as proofs-of-concept, showing the feasibility of the architecture and showcasing its effectiveness in practical scenarios.

Keywords

Architecture; Multimedia; Mobile Robots.

Resumo

Boaro, José Matheus Carvalho; Colcher,Sérgio. **Uma Arquitetura para o Enriquecimento de Fluxos Multimidia em Tempo Real com Informações Semânticas.**. Rio de Janeiro, 2023. 64p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Embora os sistemas multimídia tradicionais se concentrem na codificação e no armazenamento eficientes de tipos de mídia e suas relações temporais, a demanda atual por experiências mais ricas e personalizadas exige uma compreensão mais profunda do conteúdo semântico dessas mídias. Neste estudo, propomos a integração do processamento de nível semântico aos sistemas multimídia, enriquecendo o conteúdo com informações sobre entidades do mundo real, como objetos, ações, agentes e interpretação de linguagem. A principal contribuição desta dissertação é a apresentação de uma arquitetura para enriquecimento de dados multimídia em tempo real que usa técnicas de aprendizado de máquina para extrair representações semânticas incorporando-as ao fluxos de dados multimídia como um serviço nativo e básico. Para demonstrar concretamente a proposta, implementamos dois casos de uso que servem como provas de conceito, mostrando a viabilidade da arquitetura e sua eficácia em cenários práticos.

Palavras-chave

Arquitetura; Multimídia; Robôs Móveis.

Table of contents

List of figures

List of tables

List of codes

List of Abreviations

API – Application Programming Interface

CDVS – Compact Descriptors for Visual Search

CNN – Convolutional Neural Networks

FFMPEG – Fast Forward Moving Picture Experts Group

GPU – Graphics Processing Units

HTTP – Hypertext Transfer Protocol

ICE – Interactive Connectivity Establishment

IP – Internet Protocol

IOT – Internet Of Things

MPEG – Moving Picture Experts Group

MQTT – Message Queuing Telemetry Transport

MVIT – Multiscale Vision Transformer

NAT – Network Address Translation

P2P – Peer-To-Peer

QOE – Quality Of Experience

ROS – Robot Operational System

RTC – Real Time Communication

RTCP – Real Time Communication Protocol

SDP – Session Description Protocol

SIP – Session Initiation Protocol

STUN – Session Traversal Utilities for NAT

TURN – Traversal Using Relays around NAT

VOIP – Voice Over Internet Protocol

WASP – Wide Analytics Streaming Platform

WER – Word Error Rate

YOLO – You Only Look Once

"...without love, without anger, without sorrow, breath is just a clock... ticking"

Kurt Wimmer, *Equilibrium.*

1
Introduction

Multimedia systems encompass a wide range of services, from music and video streaming to video conferences, social networks, and virtual assistants. In the context of traditional multimedia systems, the primary challenges were mainly associated with the efficient coding of each media type, enabling efficient storage and real-time communication (FURHT, 1994). Additionally, representing, storing, and preserving the temporal relationships inherently present within and between these media types posed significant obstacles (NAHRSTEDT; BALKE, 2005).

Current systems and services, however, have brought new challenges to this scenario as more than mere representation of content alone has become a requirement. As users seek more rich and customized experiences, a need for exploring a semantic understanding of the content beyond technical aspects arises (LEW et al., 2006). Therefore, semantic-level processing becomes a natural extension of these systems' functionality, enriching multimedia content with semantic information about real-world entities, such as objects present in a scene, actions performed, and interpretation of commands, among other high-level tasks.

Incorporating this kind of knowledge into a media flow enables a more precise understanding of the transmitted multimedia content. In addition to enhancing comprehension, it opens up opportunities for various applications that can be developed based on indexed semantic information. User experience also benefits from semantic information in multimedia content, as a more sophisticated understanding of the content enables systems to react intelligently and personalize interactions, such as virtual assistants and streaming services (NECULA et al., 2018; MOBASHER; COOLEY; SRIVASTAVA, 2000).

With the advancement of multimedia research, it comprehends much more than basic multimedia services such as encoding and communication. Currently, developments in the field of multimedia involve applications such as multimedia retrieval (KONSTANTINOU et al., 2010; WAN et al., 2014; MORIKAWA; SILVA, 2012), recommendation (WANG; WANG, 2014; WEI et al., 2019), summarization (SHANG et al., 2021; SANABRIA et al., 2019). Nevertheless, efforts around media understanding and experience have gained traction regarding issues related to feature fusion (JIN et al., 2017; ZHANG et al., 2013), immersion, and quality of experience (FARIAS, 2022). Given these advances, applications that use semantic aspects of media are steadily rising,

which calls for enriching information to fulfill tasks in different environments (FU; QIU, 2012; LIDON et al., 2017; MENG et al., 2017). In this context, machine learning techniques are fundamental to enhancing data through semantic feature extraction, pattern classification, and natural language processing; thus, facilitating the use of machine learning algorithms becomes an important investigation.

1.1 Objective

In this study, we assert that current multimedia systems should be equipped with core services that enrich media flows with semantic information ready to be consumed by user applications, providing a comprehensive perception of the surrounding world or environment.

1.1.1 Main objective

As our main objective, we present a middleware architecture that enables real-time multimedia flow data enhancement, with semantic-level processing for real-time multimedia services that can facilitate applying existing or new machine-learning techniques to extract semantic representations and incorporating them into multimedia data streams as a native service.

1.1.2 Specific objectives

1. Investigating the applicability of the proposed architecture.
2. Demonstrate the applicability of machine learning algorithms for perfoming the extraction of semantic information.

1.2 Validation

To validate the proposed architecture, we have investigated its usage in two use-case scenarios. The first is a general-purpose management and monitoring application that allows the inspection, detailed analysis, and visualization of the resulting semantically-enhanced flows received by any application. In the second use case, we present an implementation of a simulated ad hoc autonomous agent in collaboration with humans that serves as a proof of concept. More specifically, we focused on the enhancement of multimedia sensory data that the robot captures in a collaborative ad hoc environment with humans.

Given the nature of the problem, the robot must rapidly adapt to changes, using the acquired semantic information for decision-making and navigation within the agent's operating environment (STONE et al., 2010). Both use cases align with the systems' requirements, where the proposed architecture can show its benefits.

1.3 Contribution

The main contribution of the present work is the proposition of an architecture for semantic enrichment directly in multimedia streams, providing a richer underlying service for developing more sophisticated applications that, ultimately, benefit from such value-added basic service and knowledge.

1.4 Dissertation structure

This dissertation is structured as follows. Chapter 2 briefly presents the main concepts used in the elaboration of this work. Chapter 3 provides an overview of the related works, and Chapter 4 describes the proposed architecture in detail. Chapter 5 outlines the use cases, followed by Chapter 6 that presents their implementation, while Chapter 7 shows some tests over that proof-of-concept implementation. Finally, Chapter 8 presents this research's overall findings and future directions.

2
Theoretical Foundation

This chapter presents the concepts explored during the development of the present work. Among them, we have: Real-time applications, real-time communication protocols, describing in details the communication protocols that we used, Ad Hoc Autonomous Agent Teams, characterizing the research field of one of the use cases of the proposed architecture. It also covers concepts like multimedia, and machine learning, describing the field, and the model's architectures that we used.

2.1
Session Description Protocol (SDP)

SDP is an Internet protocol that standardizes the representation of information about media details, transport addresses, and other session description metadata when sessions that require media information are initialized, such as multimedia teleconferences, voice-over-IP calls, and also streaming video and audio (HANDLEY; JACOBSON; PERKINS, 2006).

The general purpose of SDP is to convene information about media streams in multimedia sessions. With this information, any receiver of the SDP protocol can join the session that has been sent. An SDP session description includes the following information:

- Name and purpose of the session;
- Time that the session is active;
- The media that compose the session;
- The information necessary for receiving media (addresses, ports, formats, etc...).

Specifically related to the media composing the session and transport information, an SDP session description includes information such as the media type (video, audio, etc...), what protocol is used to transport the media (TRP, H.320, etc...), what encoding format the media is in (H.261 video, MPEG video, etc...) (HANDLEY; JACOBSON; PERKINS, 2006).

Code 1 presents what SDP consists of. Structurally speaking, an SDP session description consists of several of lines of text with the following structure:

Code 1: Session Description Protocol basic structure example.

```
1 <type>=<value>
```

where <type> must be exactly one case-signifying character and <value> a structured text that depends on the value of <type>, usually <value> can have one or more space-separated values. In the SDP session description, items marked with "*" are non-mandatory.

2.2 Network Adress Translation (NAT)

When the internal IP address cannot be used externally, either for privacy reasons or because it is externally invalid, the need for IP address translation to one that is valid and externally visible arises. Basic address translation allows hosts on a private network to transparently access an external network, and also allows access to selected local hosts from outside the network. NAT allows hosts on a private network to access hosts on an external network. In general, NAT sessions are uni-directional and are initiated by the private network (SRISURESH; EGEVANG, 2001).

2.3 Real Time Communication Protocols

Communication protocols, by definition, are a set of rules that determine how two or more entities should communicate. Such rules can define, for example, what type of message can be used to communicate, what format, and a series of other rules that determine how an entity can or should communicate with another entity (HERCOG, 2020). Therefore, real-time communication protocols are a set of communication rules between two entities that address the appropriate requirements for real-time communication. Among these protocols we can highlight: WebRTC, RTC/RTCP, SIP and MQTT.

2.4 Web Real-Time Communication

The WebRTC protocol is a disruptive communication standard that enables the transmission of real-time media data, such as video and audio, between devices on the Web using the peer-to-peer communication paradigm (SREDOJEV; SAMARDZIJA; POSARAC, 2015).

The WebRTC protocol extends the client-server paradigm by introducing direct, peer-to-peer connection between devices on the network (SALVATORE,

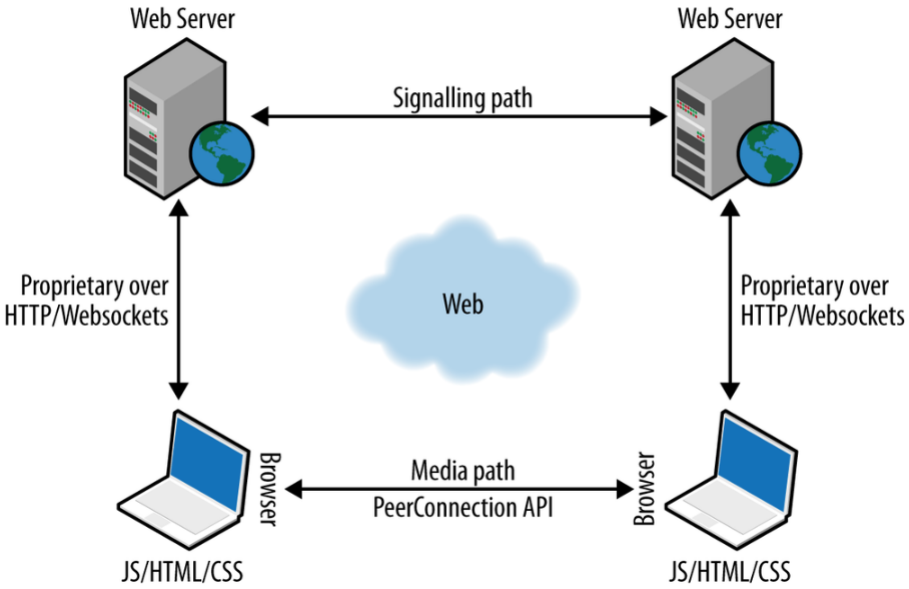


Figure 2.1: The WebRTC Trapezoid. Source: Extracted from Salvatore (2014).

2014). Figure 2.1 illustrates the communication architecture used by the WebRTC protocol.

In order for two instances to communicate using the WebRTC protocol, it is initially necessary to exchange information about the IP addresses and ports where the data will be negotiated. The WebRTC protocol does not define how this information should be exchanged, leaving the choice of technology to be defined by the programmer.

Initially, the client sends a message via WebSockets to the Signaling Server with SDP information, which responds by sending its own SDP. To allow devices connected to the Internet through a NAT router to make a direct connection, it is necessary to use the STUN/TURN ("*Session Traversal Utilities for NAT*" and "*Traversal Using Relays around NAT*") protocols. With the client's STUN/TURN information present in the received SDP package, the ICE (Interactive Connectivity Establishment) candidates are negotiated, representing the IP addresses and ports that will be used. The ICE candidate chosen is the one with the best connection, and transmission speed. Finally, the WebRTC connection is established and media information can be negotiated between the instances (SREDOJEV; SAMARDZIJA; POSARAC, 2015).

2.5 Real-Time Applications

Real-time applications are those that have the correctness of their behavior dictated not only by the logical correctness of their computation,

but also the physical instant in which the results were produced. Generally, applications that require real-time processing are control applications, such as production automation systems, where time is crucial for the correctness of the actions to be taken in this process (GOTZHEIN; GOTZHEIN; WHEELER, 2020).

Real-time applications have a number of specific requirements regarding performance, reliability, assurance, and synchrony issues. Performance requirements are responsible for aspects related to resource consumption and latency. Reliability addresses the issues of packet loss and possible transmission failures. Assurance involves the degree of commitment of the application, for example in applying the best effort or a deterministic guarantee. Finally, synchrony refers to the application's coordination accuracy regarding competing activities (GOTZHEIN; GOTZHEIN; WHEELER, 2020).

2.6 Ad Hoc Autonomous Agent Teams

The Ad Hoc Teamwork (AHT) research field aims at designing agents that can collaborate with new teammates to solve a common task without prior coordination (NEVES; SARDINHA, 2022). In AHT, agents should be capable of cooperating on the fly with other agents (either virtual agents or humans) without pre-defined protocols (MIRSKY et al., 2022).

To better understand the Ad Hoc Teamwork problem, Stone et al. (2010) made the following analogy: a tourist in a country with a different language encounters a bicycle accident while walking around a park. Several people approach each other to help (agents) without knowing each other previously, without even speaking the same language, but with the same objective, to help the injured victim. In this scenario, the knowledge and skills about the problem that each person has come into play in a coordinated way; one person looks for a doctor or policeman because she has the ability to run, and another tries to contact the ambulance to call for help, while another checks the vital signs of the victim. Finally, the individuals, even without prior knowledge of each other, coordinate to solve the problem.

An ad hoc agent must have the ability to adapt to the context of the task. Using the example above, if one of the agents knows how to perform first aid procedures, it should perform it, but if there is another agent who is a doctor, this characteristic must be identified and the role of the initial agent must be readjusted to the current state of the task. When applied to the field of autonomous agents (robots), such an agent should have the ability to adapt to the context of the task in runtime, through observations of the environment

and without prior coordination, since the other agents, autonomous or not, have no priori knowledge of each other.

According to Melo e Sardinha (2016), in an AHT setting, there are three main challenges: task identification, teammate identification, and planning. Along with those challenges, Mirsky et al. (2022) stated that there are four main subtasks that have to be fulfilled to achieve AHT: Knowledge Representation, Modeling Teammates, Action Selection, and Adapting to Changes.

The first subtask, Knowledge Representation, requires a representation of the domain knowledge. This includes information about the environment, its capabilities, and potential teammates. The representation of domain knowledge strongly influences the solution approach used in the other subtasks. With our proposed architecture that promotes real-time multimedia data enhancement, we intend to provide structured information to the ad hoc agent to help with the Knowledge Representation subtask, in a way that the agent can identify the task at hand and the teammates, and start planning its actions.

2.7 Multimedia

The definition of multimedia, according to Agrawal (2013), is the combination of image, text, sound, video and animation. In simpler terms, it is the combination of more than one media. Multimedia is a means of communication used to inform the user in a variety of ways using digital or electronic devices. Today, multimedia applications play an important role in various sectors of human life, with applications ranging from schools to doctors' offices (AGRAWAL, 2013).

Multimedia can be divided into two main forms, linear multimedia and non-linear multimedia. In linear multimedia, information or multimedia elements are presented sequentially, such as a slide show, where each slide appears sequentially. In non-linear multimedia, the multimedia components are not presented sequentially or in chronological order, in which case non-linear multimedia programs lack user interaction (AGRAWAL, 2013). Figure 2.2 illustrates the types of multimedia applications.

The complexity of multimedia applications and systems usually encompasses all components of a computer system. Given the nature of multimedia data, systems of this type require good processing power for implementing media coding, storage and formatting software. In general, multimedia systems need to support new data types, real-time scheduling and fast interrupting, also providing high storage capacity and fast disk access and file transfer. There-

fore, the multimedia area is concerned with the development of techniques and algorithms that facilitate the aforementioned problems, through compressions, transmission protocols and optimized databases for multimedia data retrieval, for instance (FURHT, 1994).

2.8 Machine Learning

The term machine learning was coined by researcher Arthur Samuel, to define the field of study related to the execution of tasks in an automatic way by computers without them having been explicitly programmed to do so (SAMUEL, 1959). More recently, with the advancement in the area, other more refined definitions have emerged, such as the one dated by Tom Michel, which states that "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E". Machine Learning is currently linked to several fields of study, among them, we can mention psychology, neuroscience, statistics, and artificial intelligence (ALZUBI; NAYYAR; KUMAR, 2018).

Among the problems that machine learning aims to help, we have problems related to classification, where the algorithm must define a truth value, true or false, for a given event. This is the case, for example, in image classification, audio classification, scene classification, among other applications.

In addition to classification problems, machine learning can be applied

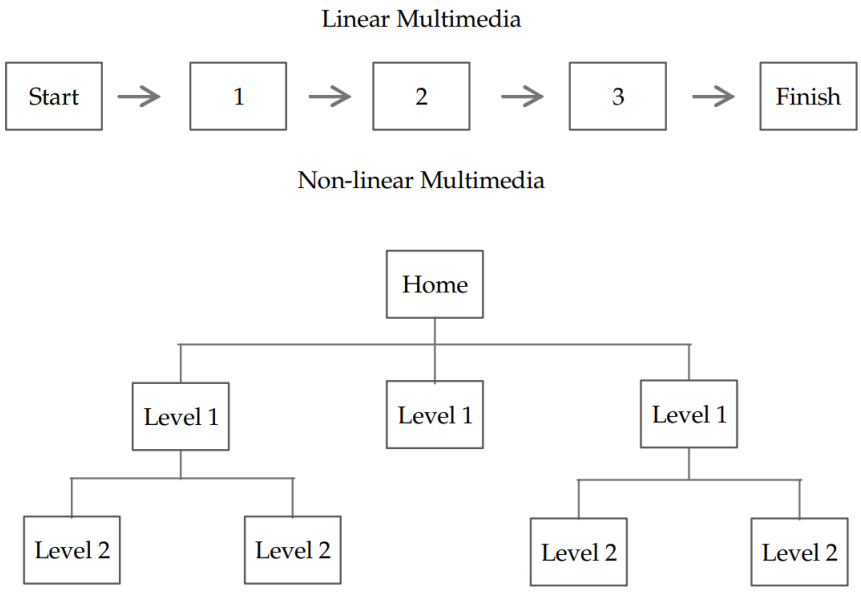


Figure 2.2: Linear and Non-linear Multimedia. Extracted from Agrawal (2013).

to anomaly detection problems, which, based on data, can identify pattern changes. As an example of such problems, we have credit card fraud detection, insurance fraud, mechanical failure detection, among other activities that have a strong pattern in the distribution of data (ALZUBI; NAYYAR; KUMAR, 2018).

Regression problems can also be solved using machine learning, which consists in performing the prediction of a continuous value based on the received data, such as temperature prediction, real estate value prediction, product price prediction, thus, generally used in problems that have as a question “how much" (ALZUBI; NAYYAR; KUMAR, 2018).

2.8.1 Neural Networks

Artificial neural networks are mathematical models structurally inspired by a tangle of biological neurons (KOVÁCS, 2002). From this inspiration was created the precursor to what are now the artificial neurons used by most current neural networks, the Perceptron. The Perceptron is a basic artificial neuron that from several binary inputs returns only a single output. This output is computed by multiplying the input values by weights, that weights have the role of addressing the importance of a given input value to the output (NIELSEN, 2015). Thus the binary output is determined when the weighted sum $\sum_j w_j x_j$ is less or greater than a given threshold (Equation 2-1).

$$output = \begin{cases} 0 & if \sum_j w_j x_j \leq treshold \\ 1 & if \sum_j w_j x_j > trehshold \end{cases} \tag{2-1}$$

The first models of artificial neural networks were developed based on the grouping of several perceptrons in a layered architecture, the Multi-layer Perceptron. In Figure 2.3, the first column of perceptrons, or first layer, makes simpler decisions just by weighting the input values, while the second layer makes more complex and abstract decisions, since it weights the output of the previous layer, so in a possible third layer, the decisions made would be even more complex. This type of information transport, where the output of one layer serves as input for another layer, makes this type of neural network a “feed-forward" neural network. This means that there are no repeating loops within the network, information is always passed forward and never backward (NIELSEN, 2015).

Moving the threshold to the other side of the inequality, and changing its notation to bias, we have that

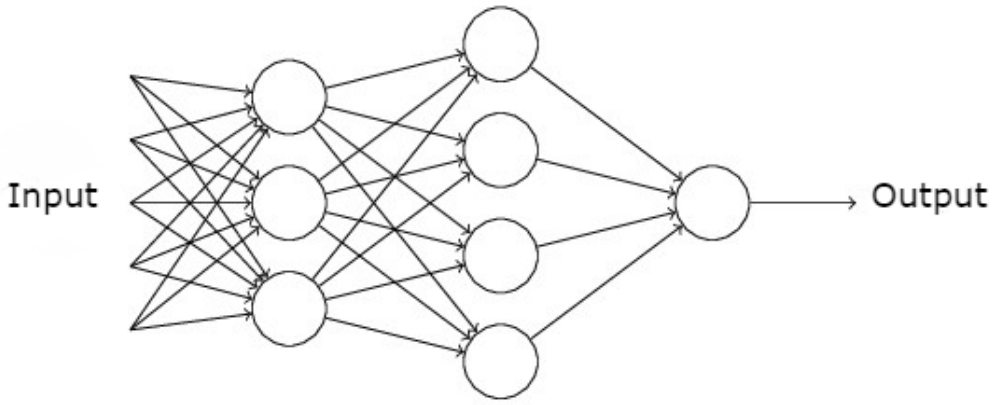


Figure 2.3: Architecture of a Multi-layer Peceptron Network. Source: Nielsen (2015).

$$output = \begin{cases} 0 & if \sum_j w_j.x_j + b \leq 0 \\ 1 & if \sum_j w_j.x_j + b > 0 \end{cases} \tag{2-2}$$

The bias now simply takes the role of a mechanism that measures how easily the perceptron is activated (output 1). The higher this value, the easier it is for that perceptron to get a 1 value in its output (NIELSEN, 2015).

To get to the artificial neural network models we have today, in terms of task complexity, the perceptron had to evolve. The weighted sum alone was no longer enough to solve some more complex problems. With this, learning algorithms began to be employed to perform the automatic search for the weights and biases of an artificial neural network (NIELSEN, 2015).

Then the Sigmoid artificial neuron was developed, it arises from the need that small changes in the values of weights and biases be reflected in small changes in the output, something that, for the form in which it was developed, could not be achieved by the perceptron. Unlike the perceptron, Sigmoid artificial neurons can receive inputs with floating point values instead of only binary inputs, moreover, their output is now no longer denoted by $\{sum_j w_j x_j + b$, but rather $\sigma(\sum_j w_j.x_j + b)$, where σ is the symbol assigned to the sigmoid function (Equation 2-3). The output of a sigmoid artificial neuron represents probability, unlike the output of the (NIELSEN, 2015) perceptron.

$$\sigma(z) \equiv \frac{1}{1 + e^{-z}} \tag{2-3}$$

From that point on, there was an evolution in the complexity of the problems that could be solved through artificial neural networks; networks of the most diverse architectures were developed, with more and more layers and different relations between them. Other activation functions, such as the ReLU (NAIR; HINTON, 2010), also started to be used. Newer neural network

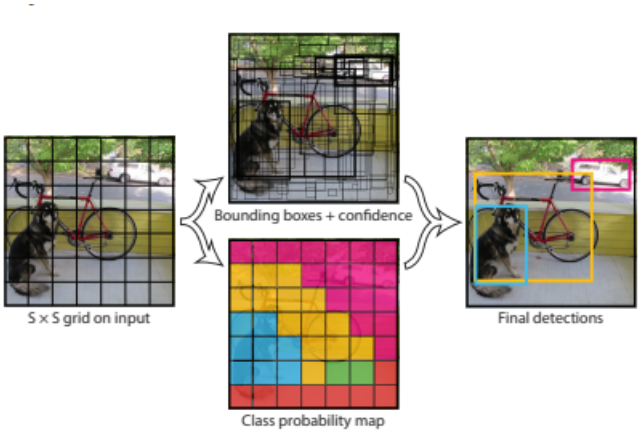


Figure 2.4: YOLO model design. Source: Extracted from Redmon et al. (2016a)

architectures, called deep neural networks, are characterized by the grouping of several layers

2.8.2 YOLO

Developed for the task of object recognition, the YOLO (You Only Look Once) (REDMON et al., 2016a) neural network is composed of a convolutional neural network that performs both bounding box prediction, as a regression problem, and probability prediction of classes. One of the main advantages of using YOLO for object detection is its processing speed, outperforming models such as FAST-RCNN (GIRSHICK, 2015), which in the context of the present work is essential.

Initially, the input image is divided by a grid of size SxS, where upon the existence of an object within a grid cell, it is responsible for detecting that object. Each grid predicts B bounding boxes and a confidence score for each of them. The confidence score corresponds to the IoU between the predicted bouding box and the truth value for the detected object. At the end, 5 values are predicted: x, y, w, h and the confidence score, where (x, y) are the values corresponding to the coordinates of the center of the bouding box relative to the grid cell boundaries, and (w, h) the width and height coordinates relative to the entire image. Finally, in addition to these 5 values, C, the conditional probability of class (REDMON et al., 2016a), is also predicted. Figure 2.4 illustrates how YOLO's model acts on an image.

Over time, YOLO has been improved to more robust versions (REDMON; FARHADI, 2016; REDMON; FARHADI, 2018; BOCHKOVSKIY; WANG; LIAO, 2020) addressing possible limitations. In the present work, we use YOLO in its version 5 (JOCHER et al., 2022), which, in its simplest

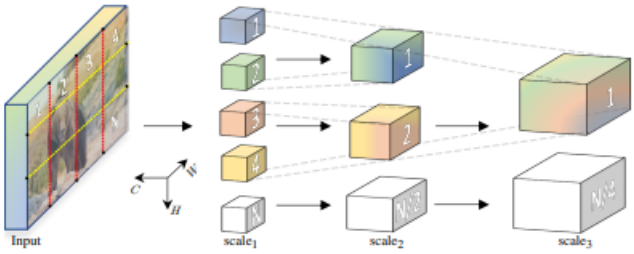


Figure 2.5: MultiScale Vision Transformer processing flow. Source: Extracted from Fan et al. (2021).

version has only 22 mb, 90% more modest than its previous version (KARTHI et al., 2021), being written natively in PyTorch, making it easy to adapt and use. It is worth mentioning that its structure and characteristics are similar to version 4, however, as mentioned, with more flexibility in the size of the models and also a better preparation in the input data (JIANG et al., 2022).

2.8.3 MultiScale Vision Transformer

Proposed by Fan et al. (2021), the Multiscale Vision Transformer (MViT) neural network architecture is based on the idea of hierarchical multi-scale features in conjunction with vision transformers to perform image and video recognition. The authors propose that the use of hierarchical multi-scale features can be beneficial for transformer models within a range of computer vision related activities due to the fact that such features are based on vision principles (FAN et al., 2021).

Figure 2.5 illustrates the processing of an image by the MViT architecture. As we can see, MViT has several stages, and each stage has several scales. Starting with the initial image resolution and small dimension channels, throughout the network architecture, the channels are hierarchically expanded while the spatial resolution is reduced, creating a pyramid of feature activation in a hierarchical multi-scale manner and connecting these principles with the transformer architecture (FAN et al., 2021).

An important feature of MViT is the ability to consider the spatio-temporal context during classification, making it ideal for video processing. The authors demonstrated that in other transformer-based architectures the order of the frames did not significantly impact the final result, meaning that these architectures did not consider the temporal context between the frames (FAN et al., 2021).

In the present dissertation, the MViT model was used in its version 2 (LI et al., 2021), which brings with it improved related integration of

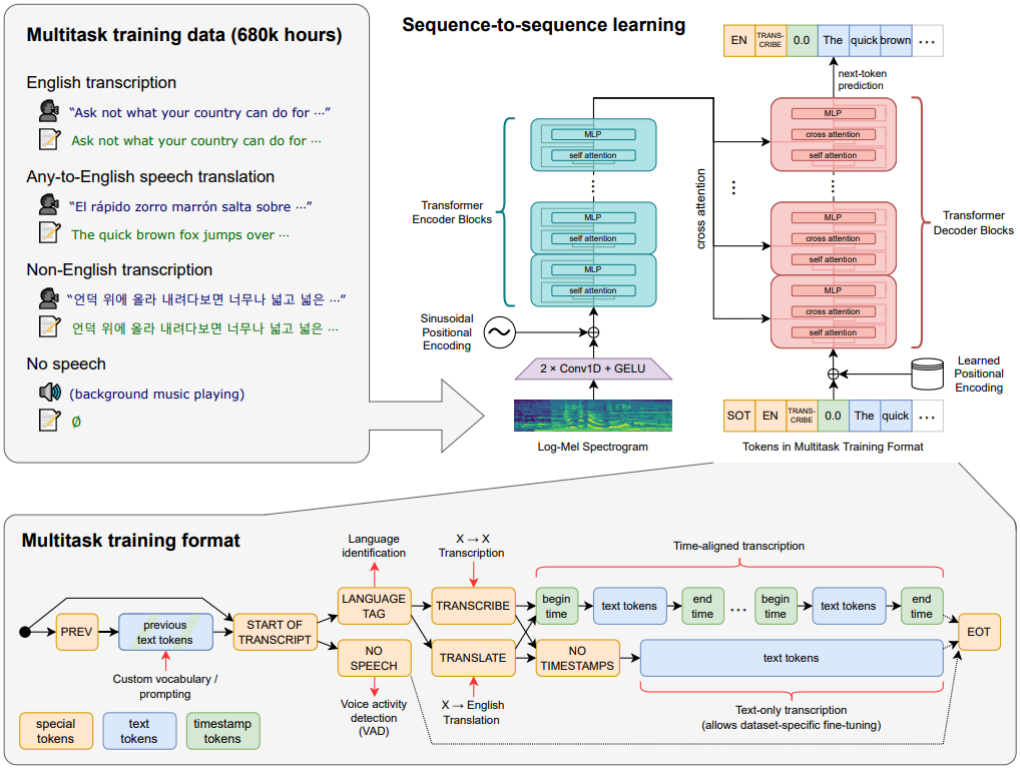


Figure 2.6: Overview of Whisper approach. Source: Extracted from Radford et al. (2022)

relative positional decomposable embeddings and residual pooling connections, reaching the state of the art in three domains: image classification, object detection, and video recognition (LI et al., 2021).

2.8.4 Whisper

Developed by the American company OpenAi, Whisper[1] is the name given to the presented approach to audio transcription. As a main contribution, the work developed by Radford et al. (2022), scales the speech recognition area with weak supervision to an order of magnitude of 680,000 hours of annotated audio data, filling a gap of previous works, which used smaller databases. In addition to the magnitude, the work also focuses not only on speech recognition in English but also about 90 other languages (RADFORD et al., 2022). Figure 2.6 illustrates how the Whisper approach works.

Initially, a seq-to-seq transformer model is trained on different audio processing tasks, among them, speech recognition, speech translation and language identification. For each of these tasks, the decoder's role is to predict a sequence of tokens, which means that the same model can replace several

[1]https://openai.com/

different stages of a traditional speech recognition pipeline. In order for the model to accomplish the various tasks mentioned, a special token is added to the input specifying the task to be performed, or the classification target (RADFORD et al., 2022).

Based on the architecture, the Whisper model has 5 sizes, ranging from tiny to large, containing 39 million and 150 million training parameters respectively. At the end of the training, the authors demonstrated that the proposed solution obtained a superior performance to other models for the tasks related to audio processing, not only in English, but also multi-lingual. Specifically for the speech recognition task, the proposed approach obtained the values of 3%, 4.2% and 4.3% of Word Error Rate (WER) for English, Spanish and Portuguese, respectively (RADFORD et al., 2022).

2.8.5 ROS

ROS is an open-source library that provides support in developing applications in the field of robotics, facilitating collaboration and software reuse among researchers and developers. With a reasonable potential of processes executing at runtime on different hosts, a system built using ROS is based on peer-to-peer connection, thus avoiding unnecessary data traffic over the network. The ROS tool is multilingual, in its first version supporting the programming languages: Python, C++, Ocatave and LISP (QUIGLEY et al., 2009).

ROS has four main foundations: nodes, messages, topics, and services. Nodes are processes that perform a computation. Each node within a ROS application can be thought of as part of a piece of software, so multiple nodes can run at the same time, and they can communicate with each other as a directed graph. A node can communicate with another node through messages (QUIGLEY et al., 2009).

A message, in the context of ROS, is a strongly typed data structure, supporting primitive data such as integers, floats and booleans, as well as arrays or sets of arrays being composed of other messages, for instance. A message to be consumed by another node needs to be published in a topic. Thus a node that has an interest in the behavior of another node, must subscribe to a topic to consume such messages. It is possible that more than one node can post to the same topic and the nodes are not aware of the existence of another node, they only have access to the topics (QUIGLEY et al., 2009). Figure 2.7 shows a simple communication pipeline between nodes.

The publisher/subscriber paradigm is not appropriate for synchronous

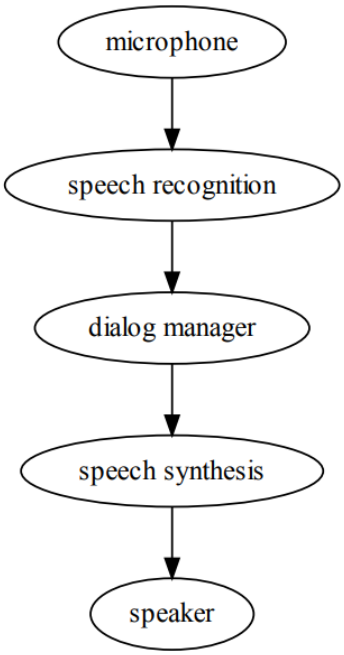


Figure 2.7: Simple ROS node communication pipeline. Source: Extracted from Quigley et al. (2009).

transactions. To overcome this problem ROS has a service implementation, which similar to a web service, restricts the name of the service and the response, ensuring a unique name and synchronous processing (QUIGLEY et al., 2009).

With a wide range of modules, ROS allows for the separation of application, which facilitates code maintenance. Through ROS, it is possible to integrate a wide range of sensors, such as cameras, microphones, and actuators. The library also features ready-to-use algorithms for navigation, image processing, speech recognition, among other tasks. One of the important aspects of this operating system is performance. As a complex system designed for research purposes, internal tools may be slower than those implemented on other customized operating systems. As one of the objectives of this work is to process sensory data remotely and in real-time, it will not be necessary to compromise the internal processing of the robot with sensory data.

3
Related Work

Investigating the main scientific productions in the research area is fundamental for the development of an academic work, since based on other productions that have been developed in similar contexts, the researcher it is able to contextualize its onw research. This chapter will address the most relevant works found during the production of this dissertation.

For the search and selection of related works, Google Scholar, Science Direct and IEEE Explore was used, and the following keywords were searched for: Machine Learning, Enrichment, Multimedia, Stream, Middleware, and Architecture. From this, the main articles found were read and those that resembled the problem developed in this dissertation were chosen.

3.1
Multimedia Enhancement

Konstantinou et al. (2010) proposed *Priamos*, a middleware system for real-time semantic enrichment of contextual features, implemented using Web Semantics technologies to aid in developing inference-based applications. The main idea of the proposed middleware is to launch a procedure that annotates contextual information on multimedia based on appearance, using a set of specific rules. Their proposed system has three main layers: Data Acquisition, Semantic Middleware, and Application. In the Data Acquisition layer, data is collected, processed by signal processing algorithms and techniques, and then an XML document is constructed and sent via web services to the Semantic Middleware layer to apply semantic mapping rules and ontology model persistence. The system was tested in two main scenarios: a security camera application and an intelligent waiting room, demonstrating that inference-based systems have reached a mature stage and can be used to provide intelligent results.

Montagnuolo et al. (2017) presented a platform that supports the ingestion of multiple video streams, applying various scalable machine learning techniques to associate them with extracted features. These features were then compared with reference patterns stored in a database to generate descriptive meta-tags for the processed content.

The proposed system was based on the Wide Analtics Streaming Platform (WASP) and Apache Spark frameworks. It consisted of three main stages: the WASP architecture, the video processing pipeline, and the semantic enrich-

ment pipeline. The WASP architecture enabled the development of complex real-time applications, particularly for IoT and big data streaming analysis, allowing developers to focus more on the business logic of the application rather than technical aspects like architecture backbones and integration components.

In the video processing pipeline, videos were ingested and decoded using the FFMPEG library. MPEG CDVS descriptors were generated for each frame and published in a Kafka queue. The specific algorithms used for generating MPEG CDVS descriptors were not explicitly mentioned, although the authors stated that machine learning techniques could be easily integrated through the APIs provided by the WASP platform. The frames were matched with other characteristics present in the database, and when visual patterns were identified, they were cataloged with the same identification number. The results were then sent to the semantic enrichment pipeline.

The semantic enrichment pipeline received the results from the video processing pipeline and incorporated semantic information from referenced web sources, such as geolocation and construction year, into a database for subsequent queries. The data was timestamped, allowing the identification of temporal relationships among the detected patterns. Finally, the video was transcoded, and the enriched metadata were made available for user visualization.

The authors tested their system in two main use cases: video search and real-time dynamic content enrichment. The first use case involved searching for videos based on detected patterns, while the second aimed to enhance video information to create more immersive experiences with low maintenance and evaluation costs.

The framework proposed by Soares e Barrére (2018) for automatic topic segmentation in lecture videos focuses on video topic navigation. Although the basis of the study are videos, the authors use only audio, slides, subtitles, and metadata for topic segmentation. Within the methodology presented for the execution of the framework, it has a feature extraction step followed by semantic enrichment, which receives low and high-level features, such as silence detection and audio transcription, and tries to relate them to other features previously stored in a database.

Švec, Neduchal e Hrúz (2022) proposed the development of a perception module for mobile robots to process audio data and images acquired from the robot's sensors. On the server side, it proposes an architecture based on *Speech-Cloud*, a system that allows communication between a client - in this context, the mobile robot - and an inference server, with support for audio processing tasks such as speech recognition, synthesis, and understanding. The proposed

architecture uses network protocols, such as *WebSockets*, for message control and SIP/RTC protocols, such as *WebRTC*, for audio transfer (VoIP). In principle, the system was divided into two controllers: the *Speech Cloud* and the Interaction Manager. Nevertheless, its latter version has implementations of computer vision tasks, such as face detection. According to the author, the Interaction Manager runs locally on the robot, and image processing is performed using the robot's internal resources. To manage the telecommunication between the client and the server, a paid service called SignalWire[1] was used.

The module proposed by the authors aims to act as a perception module for a mobile robot. On the inference server side, part of the operational logic is also instantiated, such as databases to store and retrieve information derived from data processing and the registration of agents, objects, and tasks.

Our research proposes an architecture that enhances multimedia streams at a higher level. By leveraging this architecture, we deliver to applications video and audio streams that are already enhanced with valuable information, such as locations, agents, objects, and actions performed. This enhanced data stream significantly simplifies application development by allowing developers to focus solely on adjusting their business logic without requiring extensive low-level data processing.

3.2 Related Work Comparision

Table 3.1 shows a summary of the main characteristics of the related work. General refers to the utilization of the proposed solution, whether it has general or problem-specific applications. Real-time Processing identifies whether the work in question has as a functional requirement the ingestion of multimedia data in real time or not. Adaptable relates the work in question to the adaptation of new functionalities, such as new algorithms, data or tasks. Distributed Arch. for Distribuited Architecture, which signals whether the work in question has the layers of its architecture in a distributed way or with the possibility of being distributed. Finally Enh. Flux for Enhanced Flux, which indicates if the artifact produced is an enhanced flux, the media data along with the semantic information acquired.

Unlike previous works focusing on analyzing and enhancing multimedia content at a low-level, our approach aims to provide high-level semantic information, simplifying application development and reducing complexity.

With many similarities between our work and the one proposed by Konstantinou et al. (2010), the main difference is that the architecture proposed

[1]https://signalwire.com/

Work	General	Real-Time Processing	Adaptable	Distribuited Arch.	Enh. Flux
Konstantinou et al. (2010)	Yes	Yes	Yes	Yes	No
Montagnuolo et al. (2017)	No	Yes	No	No	No
Švec, Neduchal e Hrúz (2022)	No	Yes	No	Yes	No
Soares e Barrére (2018)	No	No	No	No	No
Ours	Yes	Yes	Yes	Yes	Yes

Table 3.1: Related work overview.

here aims to provide the enriched flow to the application layer directly, without the necessity of creating ontology models. In this way, we leave any application logic, such as the application of rules and semantics and any kind of data persistence, at the application layer, making the proposed architecture simpler and more general, being able to be adapted to several contexts, fitting the application needs. Furthermore, in Konstantinou et al. (2010), the data processing occurs before the semantic middleware layer, receiving an XML with the semantic information extracted in its Data Acquisition layer. In the architecture proposed by our work, the system receives raw multimedia streaming data and performs internal processing as part of the semantic enrichment process.

In contrast to Montagnuolo et al. (2017), our research addresses the enrichment of multimedia streams at a higher level by proposing an architecture that functions as a middleware. Our proposed architecture already provides enriched data in real-time streams, allowing developers to focus solely on adjusting their business logic. To ensure the efficiency of our proposed architecture, we suggest utilizing real-time streaming algorithms and parallel data processing architectures, including Graphics Processing Units (GPUs).

In the work proposed by Soares e Barrére (2018), the video stream received by the framework is a raw stream that will be internally processed and made available for video lecture application with topic navigation. In our work, we propose an architecture that already provides the video stream enhanced with semantic information.

Related to the architecture proposed by Švec, Neduchal e Hrúz (2022), our has a broader usage and purpose, separating the processing services from the operational logic. In this sense, besides being able to be used in the context of mobile robots, it can also be used in other contexts that require semantic information enrichment. Due to this fact, multimedia input devices, like cameras and microphones, do not necessarily need to be tied to the robot's operating system. It is sufficient for them to be directly connected to the proposed architecture. Once they possess the processed enhanced multimedia flow, the robot's operating system can make the semantic information available for internal use.

4
Architecture

The proposed architecture for multimedia semantic enhancement is depicted in Figure 4.1. It consists of four layers (Sensing, Media Flow Management, Semantic Enhancement, and Application) and a Communication Bus that seamlessly connects them. Layers are designed with a high level of abstraction, enabling them to adapt to a specific problem flexibly.

Each layer is designed to perform distinct and well-defined functions along the enhancement process. The Communication Bus allows them to be locally, remotely placed or even distributed across multiple devices according to application demands. In cases where enhancing streams involves significant algorithmic complexity and results in high computational costs, it can be advantageous to distribute or assign this workload to a machine with greater computational capabilities. The architecture remains sufficiently flexible to support such distribution and ensure the delivery of enriched information as needed. The stream process output of each layer is depicted in the arrows connecting each architecture layer to the Communication Bus.

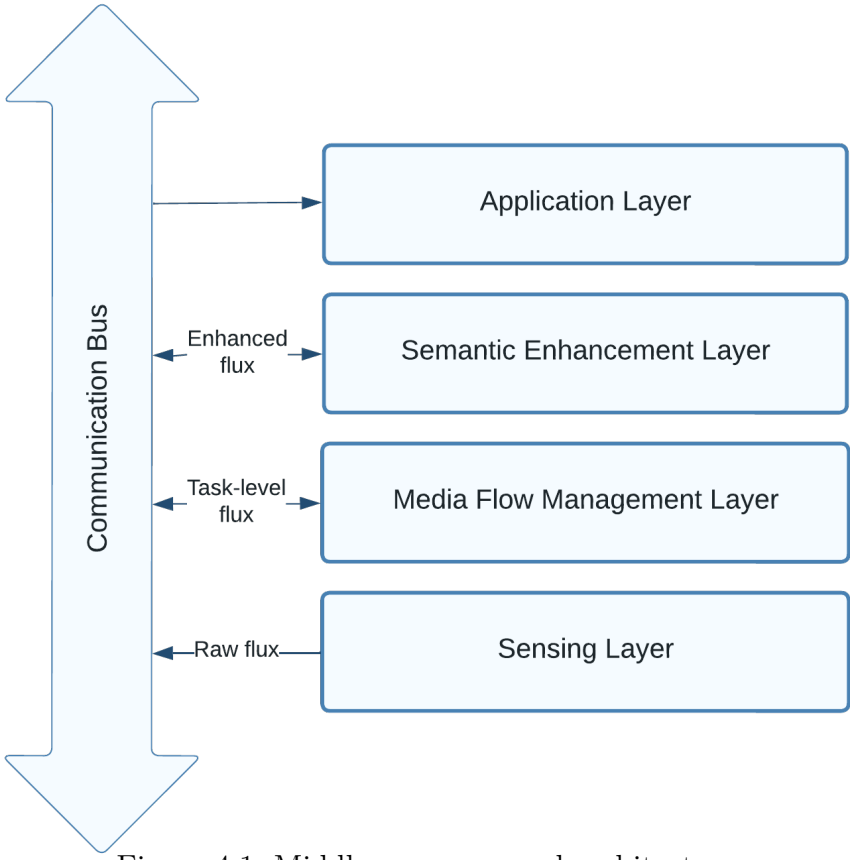


Figure 4.1: Middleware proposed architecture.

The enrichment process starts with multimedia streams being captured

by sensor devices such as cameras and microphones in the Sensing Layer. The data source can be external, transmitted remotely, or captured and transmitted locally in the instance running the middleware. In external transmissions, the Communication Bus handles the connection and communication to the adjacent layer.

Once the connection and transmission of the multimedia stream are established, the raw data stream is passed to the Media Flow Management Layer, which is responsible for the temporary storage of the received data using buffers, if necessary. The Media Flow Management Layer also performs the role of triggering the process of extracting semantic information, preprocessing the data, and preparing it for the tasks that will be executed in the next layer. This is an essential role since each semantic extraction technique can demand specific time intervals to be performed, for example, an algorithm for analyzing actions on a video.

In the next step, the flow, with the task's specific formats, is forwarded to the Semantic Enhancement Layer to initiate the extraction of the semantic information. This layer is responsible for handling the data for each semantic extraction task incorporated into the middleware. These tasks can be defined according to the application that consumes the resulting information, and depend on the nature of the data that is consumed.

Finally, the semantic information generated can be associated with the raw data flow, forming the enhanced flux, and be sent through the Communication Bus, to the Application Layer.

As mentioned before, each layer within the architecture is interconnected through the Communication Bus, serving as a conduit for the distribution of data. However, it is significant to highlight that all communication can only occur between two adjacent layers and unidirectionally. As presented in Figure 4.1, progressing from the Sensing Layer to the Media Flow Management Layer, then advancing to the Semantic Enhancement Layer, and culminating in the Application Layer.

The middleware's enhancement functionalities provide a series of facilities for the development of multimedia applications, given the provision of semantic information, without the need for the application to process the received flow. The following subsections provide more details about these layers and the Communication Bus.

4.1 Communication Bus

The Communication Bus is responsible for managing all communications among the middleware layers. Through it, data at different processing stages can be transported to be manipulated by the middleware until it reaches the Application layer.

Within this bus, transmission and connection protocols must meet the requirements associated with establishing connections and transmitting real-time multimedia streams, particularly in cases where the application or the media flow manager is not running on the same machine. The definition of these appropriate application protocols is crucial for the proper functioning of the architecture, effectively avoiding potential bottlenecks.

4.2 Sensing Layer

The Sensing Layer is responsible for initiating the whole semantic enrichment process. Its primary role involves transmitting the captured multimedia data to be enriched by the middleware and solving issues related to data coding and transmission from sensors through the Communication Bus, if necessary.

With its design as a layer and connected to the Communication Bus, it allows data transmission to be remotely performed, adding more versatility to the middleware. In this scenario, the Communication Bus enables remote connection through the utilization of real-time communication technologies. As output, the Sensing Layer delivers the unprocessed multimedia streams, named *Raw flux*, originating from sensory input sources, such as images captured by a camera device.

4.3 Media Flow Management Layer

The Media Flow Management Layer receives the *Raw flux* and, subsequently, preprocesses it producing a *Task-level flux* as output, primed for enrichment. This layer is responsible for identifying the type of data being received and preparing it according to the semantic extraction tasks performed in the upper layer. The structuring of received data streams must take into account and maintain temporal indicators, such as timestamps. These timestamps serve the crucial purpose of linking the extracted semantic data to the precise moment in time to which the data pertains.

In addition, this layer bears the responsibility of delineating settings related to the processing and storage of the data flow, which can be adjusted

depending on the requisites and tasks to be performed. It must facilitate the configuration of parameters related to trigger intervals for the semantic enrichment tasks, playing an important role in resource management. This is particularly significant as tasks that do not need continuous processing can be configured to activate at a specific interval.

Specifically, each multimedia data that is processed by the middleware necessitates a specification outlining how the Media Flow Management Layer should handle it. This is essential because the treatment of data may vary according to its inherent characteristics. For instance, consider data with temporal dependencies, such as audio and video content. Given that data is transmitted in the form of a continuous flow, it becomes necessary to create an auxiliary buffer to store each segment of the flow as it is received. In contrast, an image, that may lack temporal correlation, does not require the implementation of such a buffer.

4.4 Semantic Enhancement Layer

The Semantic Enhancement Layer receives organized media data and applies the appropriate processing tasks for each media type. This layer effectively implements all the essential tasks required to extract semantic information from the media streams. It is triggered by the Media Flow Management Layer, which forwards the preprocessed flow to each specified task.

After receiving the data handled by the media flow manager and applying specific processing algorithms to that data, the produced result is called *semantic artifact*, as illustrated in Figure 4.2, which contains the semantic information extracted from the data associated with the data flow.

The separation between Media Flow Management Layer and Semantic Enrichment Layer was designed to isolate the execution of semantic information extraction tasks from the flow management. With this, algorithms that may be added to the middleware by necessity can be more easily integrated, reducing interventions in other layers of the architecture. In this context, it can be expanded without the need for in-depth technical knowledge of the rest of the middleware layers and can be used in various contexts in which the application demands.

Specific problems determine the definition of the tasks implemented in the Semantic Enhancement Layer and may vary depending on the objectives and requirements. For example, in the context of audio streams, a list of pertinent tasks could include entity recognition, sentiment analysis, topic

classification, or any other semantic analysis task relevant to a specific domain.

Also, in this layer, the enhancement flow is further refined to generate a tuple that includes timestamp information. This timestamp serves as a clear indicator of the exact time when the data was received. The tuple also contains the type of semantic information, which indicates the task or context to which the semantic information is relevant, as well as the high-level semantic information itself.

In Figure 4.2, we show an example of the structure of the response tuple containing the semantic information that will be provided to the application Layer.

```
enhancement_object= {
    'task_1': [ts, [semantic_info]],
    ...
    'task_n': [(in_ts, fi_ts), [semantic_info]]
}
```

Figure 4.2: Response object example.

Initially, a dictionary contains an identifier of the task to which that semantic information is related, followed by its timestamp, identified by ts, and a list containing the actual semantic information, identified by $semantic\_info$. For example, in an object detection case, this list may contain all the identified objects, their bounding boxes, and the prediction confidence, leaving it up to the developer to add the necessary information to the application.

Finally, tasks with a temporal relation, for example, action detection in videos, must send the timestamps of the initial and final frames, identified by in_ts and fi_ts, respectively, together with the semantic information related to the task. Knowing this structure, the application can consume the semantic information received along with the raw data flow.

```
semantic_artifact= {
    'object_detection':
    [1694836056, [{'class':'chair','confidence':0.78},
    {'class':'refrigerator','confidence':0.98}]],
    'face_detection':
    [1694836056, [None]]
    'action_detection':
    [(1694836056, 1694836056),['walking'],
    'ner':
    [(1694836056, 1694836056),[{'agents':'Person','location':'double bench'}]]
}
```

Figure 4.3: Semantic artifact example.

In Figure 4.3, we can see an example of the response tuple containing the semantic information delivered to the Application layer.

4.5 Application Layer

The Application Layer specifies the applications that will consume the enhanced data, representing the highest level of abstraction in the presented architecture model. Moreover, it can efficiently handle the flows enhanced by the semantic enhancer.

The Application Layer must be able to receive the enriched media stream and read the semantic information that is attached to it. The stream sent should follow an information organization pattern, such as the one shown in Figure 4.2, where each task that generates semantic information should be identified by a unique identifier and its respective semantic information. Other semantic data organization patterns can be developed depending on the application in order to meet its specific requirements, as long as there is cooperation between the lower-level layers and the Application Layer. In that sense, the application must have a decoder that has the ability to read the enriched multimedia stream.

4.6 User and Roles

The middleware architecture for semantic enrichment of multimedia streams proposed in this dissertation benefits two main types of users: backend developers and application developers.

4.6.1 Backend Developers

Primarily, the function of backend developers, within the context of middleware, is to undertake the initial delineation of the activities inherent to the forthcoming application's development. After this investigation, the developers proceed to implement and integrate the corresponding solutions. These solutions are subsequently integrated into the Semantic Enhancement Layer. Within this layer, the developer outlines the algorithm and defines the semantic output that the application will utilize.

They also assume the responsibility of configuring the activation for the aforementioned tasks, aligned with the application's exigencies. This configuration is facilitated through the intermediation of the Media Flow Management Layer, where it is also possible to configure other data handling parameters.

4.6.2 Application Developers

The role of application developers comprehends the assimilation of the augmented data flow and the subsequent application of pertinent business rules coherent with the necessities of the developed application.

To exemplify this within the scope of the suggested experimental application, namely the mobile robot simulation, an enriched audio stream replete with spatial and agent-specific data reaches the Application Layer. From this data flow, the application developer deploys a decision-making logic, thereby executing the robot's navigation through the environment. Notably, since the audio data is already enriched by the time it reaches the Application Layer, the developer's primary concern is to ensure its proper utilization. The sugested use case is further deatailed in Chapter 5.

5
Use Cases

To test the proposed architecture, we have investigated its usage in two use cases. The first is a general-purpose management and monitoring application that allows the inspection, detailed analysis, and visualization of the resulting semantically enhanced flows received by any application in the Application Layer. The second use case corresponds to a more specific application-oriented scenario of a mobile robot in an ad hoc collaborative environment.

5.1
Monitoring Application Use Case

The monitoring application was specifically developed to serve as a support application that attests to the usability of enhanced flows and provides a management interface for configuring the media flow manager and communication parameters. Its graphical interface can also help users by showing a snapshot of the extracted semantic information, including detected objects from a received video track and the extracted information from an audio track, thus playing an important role in the visual evaluation. Although the primary intention of developing a graphical interface was to facilitate testing and visualization of results, it also allows for checking models' performance and identifying areas for improvement or possible errors, providing a way of monitoring and real-time evaluation. Figure 5.1 illustrates the proposed interface for the developed perception module.

With this interface, the user can choose the parameters on the left side, including the host IP and port where the architecture layers are available for applications and multimedia flows. Additionally, the user can set values for the frame interval of the object detection and action recognition models. It is also possible to define the buffer's size for storing audio data and the time interval the audio recognition model is executed.

In the lower left part of the interface, the time of execution of each task performed by the module is displayed, which can be used to compare techniques and models implemented. Information about the system operation is displayed in the upper right region of the interface. An example that can be seen is the model communication status and whether it is receiving video or audio streams.

In the middle region of the interface, the received video stream is

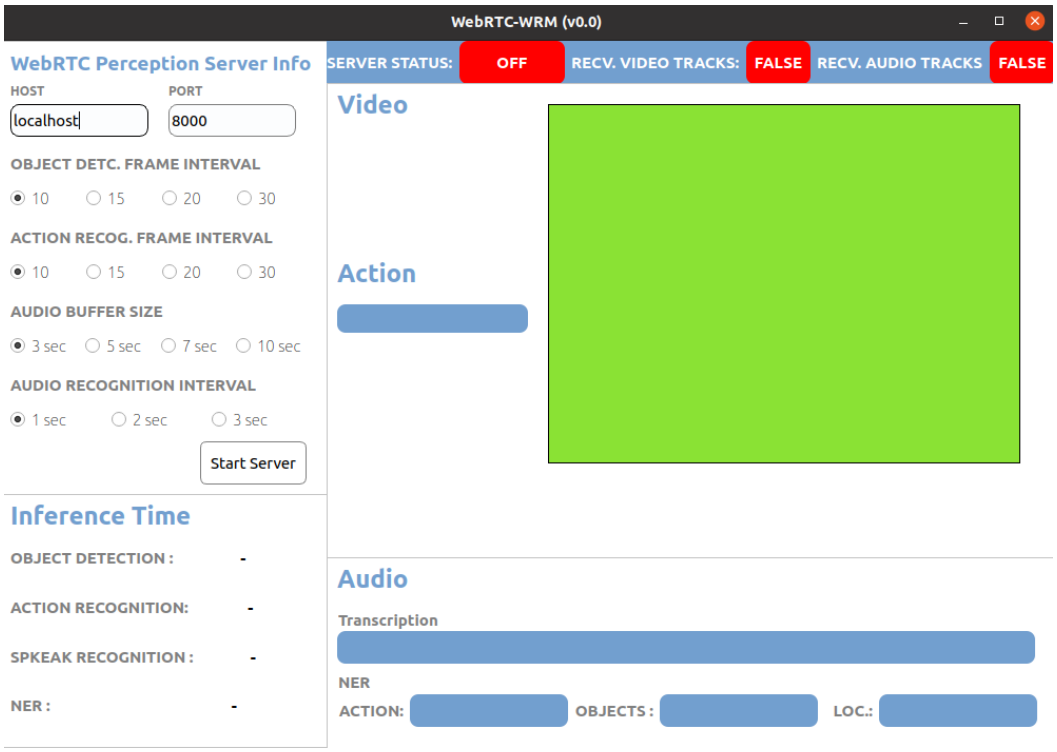


Figure 5.1: Proposed module management interface.

displayed along with the semantic information extracted. This can include recognized objects, people, and faces to facilitate the model validation for each task. Finally, in the lower right region of the interface, information about audio tasks is displayed if an audio stream is present. The displayed information includes the received audio transcription and the recognized entities in that speech, such as actions, objects, and locations.

5.2 Mobile Robot Perception Module Use Case

Our second use case involves the perception module of a mobile robot designed to work in an ad hoc collaborative environment with humans. In collaborative ad hoc scenarios, agents must cooperate with each other to complete a task without any prior coordination (MIRSKY et al., 2022). Since cooperation happens on the fly, agents have to gather information about the environment and reason over it to help their teammates achieve a common goal in real-time. Ad hoc agents play several roles in this context, like identifying the task to be executed, getting descriptions of the environment in which the task is happening (e.g., detect objects and get their location), and identifying or characterizing each teammate, among other functions (MELO; SARDINHA, 2016). All this information can be used as input for the decision module of the agents, and their responses should happen accordingly in real-time.

In such a collaborative scenario, the agent (in our use case, a mobile robot) must be designed to function inside unpredictable and rapidly changing environments, making it central to obtain accurate real-time responses and information to adapt to new conditions as they arise – a problem to which the multimedia processing module is intended to be suited. In this context, mobile robots need to understand their surroundings to navigate and make decisions in real-time due to the dynamic nature of the problem, where agents and objects can be constantly changing.

This use case aims to take the multimedia streams available on the robot and its environment and simulate the utilization of the resulting enhanced streams to assist in its navigation and decision-making processes. By applying the proposed architecture in this context, we can validate its functionality, from the transmission of the multimedia stream to the delivery of the enhanced stream to the application.

Specifically, in our work, ASTRO is the mobile robot to be simulated (MELO et al., 2019). ASTRO is a mobile platform created for multi-modal interactions between humans and robots and has a set of lasers used for autonomous navigation and obstacle detection. Equipped with an LCD screen and a rotational head, ASTRO can communicate with human agents through facial expressions and speech. To detect specific objects in an environment, such as balls in an enclosed environment, ASTRO has RFID sensors and a basket for collecting objects. In addition to these sensors, the robot has a camera and a microphone.

The proposed application will be initially conducted in a simulation of a collaborative Ad Hoc mobile robot using ROS (Robot Operating System) library. In our project, we used ROS 2, an evolution of its previous version, to fulfill modern mobile robotics requirements in new domains and production environments (MACENSKI et al., 2022). In this use case, it is possible to test the mobile robot's navigation through a controlled environment using the speech of a human agent in the environment as a guide. By using speech identification, transcription, and identification of entities such as location and agents, ASTRO can navigate the environment correctly to safely and reliably complete its task.

In its current state, ASTRO is prepared to use only audio streams in its decision-making and navigation activities. Nevertheless, the main objective of applying the proposed architecture in a realistic scenario can be achieved by submitting the stream coming from ASTRO's microphone into the developed multimedia stream enhancement model.

Moreover, we have also decided to investigate the distribution versatility

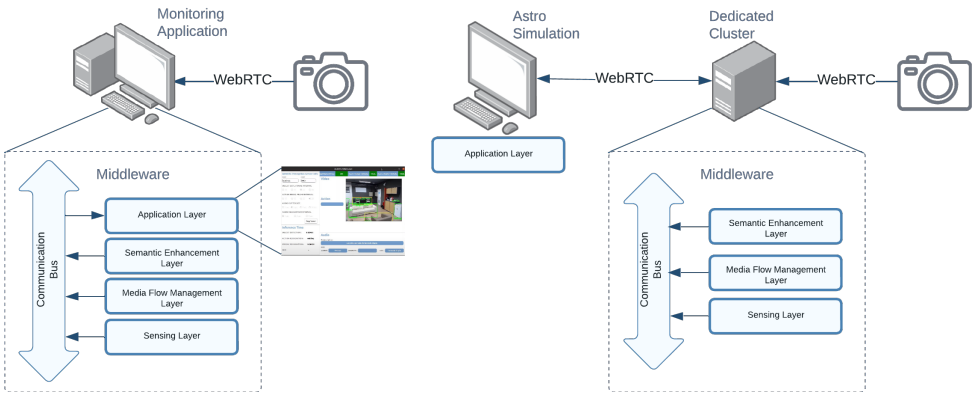
allowed by the architecture, by exploring a case in which the stream is remotely enhanced and then transmitted to the mobile robot.

In addition to exercising the distribution availability of the architecture, by providing the already enhanced audio stream to the robot, we eliminate the need to spend its internal processing power on activities such as speech processing and entity-recognition models. This ensures that the robot's resources remain readily available, allowing the navigation and decision-making processes to occur with minimum latency.

6
Implementation

Our implementation covers both the monitoring application and the ad hoc robot application. The main difference between the two approaches is that in the monitoring application, all layers of the architecture run on the same location; in contrast, the ad hoc robot application, due to its requirements, has layers running distributed, as the processing capabilities of the robot can influence real-time information processing.

Figure 6.1, illustrates the implementation of the proposed architecture applied to the two use cases. 6.1.a, the application layer is instantiated on the same device as the other layers. 6.1.b, the application layer is instantiated on a remote device, simulating the mobile robot. As the monitoring application has no specific requirements, we utilized the requirements of the mobile robot scenario as a basis for bowth applications.



(a) (b)

Figure 6.1: Diagram of the proposed architecture implementation based on (a) monitoring and (b) Ad hoc robot simulation's use case.

The middleware was implemented using Python[1] language version 3.9. For the first use case, the application was developed using the PyQT[2] library, which allows interfaces to be created. For the second use case, the ROS library was used in its distribution for python.

6.1
Communication Bus

For the robot application scenario, we considered the hardware processing limitations that could occur in a real-world scenario to choose the best way to

[1]https://www.python.org/
[2]https://www.qt.io/

implement it. As ASTRO does not have dedicated hardware acceleration, such as a GPU, it can be challenging to generate real-time information, given the complexity of the implemented models for performing tasks in the Semantic Enhancement Layer.

Therefore, for this particular use case, there is a necessity for the application to run on a different instance from the rest of the implemented layers. In this context, data must be transmitted to the application, using a real-time data transmission protocol to minimize information delivery time.

In this case, the application remotely connects through the Communication Bus using web-based real-time communication protocols, such as WebRTC.

6.1.1 Web Real-Time Communication

The Web Real-Time Communication (WebRTC) protocol was chosen for communication and data transmission between the remote application and the Communication Layer to meet the requirements for our use case. WebRTC, developed by Google, emerged in 2011 and has since been extensively used in various applications that demand real-time transmission of multimedia data, such as video conferencing and screen sharing. It enables the transmission of audio, video, and data without the need for additional plugins while incorporating encryption mechanisms to ensure the security of the transmitted information and safeguard users' privacy (RAHAMAN, 2015).

The main advantage of using the WebRTC protocol for multimedia data transmission relies on its low transfer latency since it operates in a peer-to-peer manner, where the connection between the client and the server is made directly without any intermediary. On the other hand, one of its main disadvantages is the need for a high-quality internet connection (LORETO; ROMANO, 2012).

In our test cases, the choice of WebRTC has shown itself to be a very adequate one: Widely compatible and easy to integrate, the protocol aligns directly with the functional low-latency requirement for the proposed robot's use case.

6.1.2 Establishment of P2P connection

For two instances to communicate using the WebRTC protocol, it is initially necessary to exchange information about the IP addresses and ports where data is negotiated. The WebRTC protocol does not define how this

information should be exchanged, leaving the choice of technology to be determined by the programmer. In this work, we implement a Signaling Server to exchange information and use WebSockets to establish connections.

6.1.3 Data Transmission

After the P2P connection establishment step, the client can transmit data. This enables the client to consume the enhanced information processed and made available through the architecture. Additionally, the client can simultaneously transmit audio or video data. Furthermore, in addition to establishing multimedia data stream channels, a separate data channel is also established, which will be used for server-to-client information delivery.

6.2 Sensory Layer

The Sensory Layer was implemented using video capture from a webcam and audio capture from a headset. In this sense, the sensory layer captured the media files from both devices and, based on predefined video and audio encoders, transmitted the data flow to the subsequent layers of the architecture via the Configuration Bus. To try to simulate the environment of use case 2, the mobile robot, as best as possible, the sensory layer was developed using the ROS library, where a node was created that was responsible for the entire process of capturing, coding and transmitting the data flow, imagining that in the real application, the capture devices will be on board the ASTRO robot.

6.3 Media Flow Management Layer

In the implementation of the media flow manager, we focused specifically on handling audio and video streams, as it aligned perfectly with the requirements of the proposed use cases. This layer is designed to receive the multimedia streams from the Sensory Layer. Given the nature of the use cases, media streams can be directly managed by the robot's operating system and transmitted by it.

Based on the data that we worked with, we implementent two flow handlers, one for video data and the other one for audio data. The video flow manager performs ingestion of the video data, orchestrates the activation of each task based on the received frame interval, and stores a buffer with the received data for tasks that require temporal analysis, delivering the organized video data to be further processed by the semantic enhancer. Similar

to the video flow manager, the audio flow manager also performs the role of ingesting the audio stream, organizing and triggering tasks, with some additional configurations such as the number of transmitted channels, sample rate, and audio format.

These are necessary for correctly reading and transforming audio data before applying the semantic enhancement. In our implementation, this entire process occurs asynchronously for both media types, allowing the media flow manager not to get blocked while waiting for the semantic enhancer to finish.

6.4 Semantic Enhancement Layer

To comply with the requirements of the proposed applications, it was necessary to implement algorithms that solve computer vision and audio processing tasks that take place in the Semantic Enhancement Layer.

Once the semantic enhancer extracts information from both audio and video streams, the semantic information is structured along with times stamps, and transmitted to the application via the Communication Bus.

The following tasks were implemented based on the needs of our proposed use cases. Firstly, the semantic enhancer can address a comprehensive range of computer vision problems.

All the models used were developed using the PyTorch[3] framework in its 2.0 version. PyTorch is a library that facilitates the development of machine learning models optimized for GPU and CPU, making it easy to use and develop machine learning algorithms

6.4.1 Agent Detection

The robot works on collaborative tasks with humans, and it is crucial that it can detect agents also operating in the environment. More traditional methods have already been proposed to solve this task. Such methods range from the use of visual descriptors through the identification of edges, contours, and lines (SCHLEGEL et al., 1998) to the application of image matching (GUPTA et al., 2016). However, the solutions mentioned do not perform satisfactorily in real-time applications, given the algorithmic complexity of each solution, which impacts the robot's functionalities. To address the aforementioned problems, solutions based on deep neural networks have been developed (BRUNETTI et al., 2018), and deep neural networks have proven to be a viable and robust alternative in human agent detection (ALGABRI; CHOI, 2020).

[3]https://pytorch.org/

6.4.2 Face Detection

Recognizing who the agents operating in the environment are is one of the pillars for solving collaborative ad hoc problems, playing an essential role in defining the task to be executed by the robot and impacting its reliability (MELO; SARDINHA, 2016). Therefore, face detection is a preliminary step in recognizing agents in robotic systems, such as security and domestic services (CEBOLLADA et al., 2021). The MTCCN model (ZHANG et al., 2016), a multitask cascaded convolutional network, was used for face detection in our implementation. The architecture of this network consists of three cascaded convolutional neural network models designed to predict face and landmark locations. The implementation used was provided through the FacenetPytorch library.[4]

6.4.3 Action Recognition

Action recognition involves understanding activities and gestures (OLATUNJI, 2018). Like agent recognition, action recognition also plays a fundamental role in the robot's decision-making process. Recognizing the actions being performed in a collaborative environment is vital for the robot to define how to collaborate (MELO; SARDINHA, 2016). At the time of this writing, approaches based on self-supervised learning and cross-attention are the ones with the best performance on existing databases for the task of action recognition (WANG et al., 2022). We have chosen the MultiScale Vision V2 (LI et al., 2021) model for our implementation; this second version changes the attention pooling layer, improving the metrics obtained in the task, and is the state-of-the-art for action classification in the Kinect 400 (KAY et al., 2017), Kinect 600 (CARREIRA et al., 2018), and Kinect 700 (CARREIRA et al., 2019) datasets, surpassing its previous version and convolutional neural networks. In the present work, we used the pre-trained model with 400 classifiable actions (KAY et al., 2017).

6.4.4 Object Detection

The task of object detection is related to the ability to identify an object in the environment and to classify it specifically, such as the identification of cars, trucks, and bicycles performed by autonomous vehicles (CEBOLLADA et al., 2021). In an ad hoc context, such detection provides information about

[4] https://github.com/davidsandberg/facenet

objects of interest present in the environment, such as in robots used for hazardous material extraction, where object detection can help identify such materials. In general, CNNs are used to perform this task, given their ability to distinguish and recognize objects (ZHAO et al., 2019). In robotics, object detection also plays an essential role in estimating the position of objects from various viewpoints (GARG et al., 2016). The YOLO (REDMON et al., 2016b) network in its V5 version was used for object detection. One of its main advantages, besides its robustness in detecting objects and being less prone to false positives in the background than other detection models, is its fast inference speed. It can process up to 45 frames per second, and in smaller versions, it can reach up to 155 frames, making it an interesting option for real-time object detection problems. The pre-trained version of YOLO on the COCO (LIN et al., 2014) dataset was used.

6.4.5 Speech Recognition

Speech recognition aims to identify and transform audio signals into text (YU; DENG, 2016). In the context of this proposal, the application of speech recognition methods aims to identify explicit commands given by the agent, facilitating human-machine interaction. Currently, state-of-the-art models based on deep neural networks are pointed out in the literature due to their effectiveness in processing raw audio signals, surpassing statistical methods proposed during the initial development of the problem (YU; DENG, 2016). Another crucial point in favor of using deep neural networks is the dynamic nature of the problem. With the space-time processing capacity provided by deep learning algorithms, the problem of speech recognition can be modeled considering the dynamic nature of natural language, being an adequate solution for applications that involve speech processing, such as mobile robots in ad hoc environments (RIBEIRO et al., 2021). The Whisper model, proposed by OpenAI, was used for speech recognition. It is a transformer sequence-to-sequence model trained on a large dataset and can be used for various purposes, such as multilingual speech recognition, translation, and language identification(RADFORD et al., 2022).

6.4.6 Entity Recognition

Entity recognition aims to identify mentions of predefined semantic entities in texts, such as people, locations, and organizations (LI et al., 2020). More classical techniques, such as manual rule creation (SEKINE; NOBATA,

2004), unsupervised approaches (NADEAU; TURNEY; MATWIN, 2006), or classifier-based methods such as decision trees and support vector machines (SZARVAS; FARKAS; KOCSOR, 2006; MCNAMEE; MAYFIELD, 2002), cannot find complex features as deep neural networks do. This ability has allowed deep neural networks to dominate the field, achieving state-of-the-art performance in the task (LI et al., 2020). In the context of mobile robots in a collaborative ad hoc environment with human agents, entity recognition works together with speech recognition (RIBEIRO et al., 2021). After the speech is recognized, it is necessary to process the text for information, such as actions, locations, and objects, which are essential for the robot's decision-making process. We used a customized pre-trained model proposed by Ribeiro et al. (2021) for entity recognition. The model was specifically trained to parse communication between humans and a mobile robot, particularly in an ad hoc environment. It was trained to recognize entities given commands in Portuguese, the linguistic context in which the mobile robot, ASTRO, is applied. This model can identify entities such as locations and agents, helping the robot navigate the environment with this information. It achieves an accuracy of 77.77% in correctly identifying entities in the received commands.

7
Use Case Assessments

After defining the use cases and implementing the proposed architecture model for semantic enhancement of real-time multimedia streaming, tests were performed to verify the functionality and applicability of our implementation in these use cases.

7.1
Monitoring Application

For the first use case, tests were carried out in a controlled environment. The environment was equipped with a camera and microphone, and when these were coupled to the middleware via the sensory layer, the developed application received the enriched data stream. In the tests carried out, a human agent walked around the environment interacting with objects and performing actions, as well as speaking locations. With the semantic information passed along in the data flow, the application correctly consumed this information and presented it through the developed interface

In Figure 7.1 we can see the use case's running interface. Through it, we can observe indicators for the audio and video flows, as well as the transmitted image and the objects, agents, and actions performed, identified using computer vision models. Additionally, we can see the results of the audio transcription and NER model executions. Finally, the indicators in the lower-right corner show metrics related to the response time of the implemented model for each task.

Through this application, the extracted semantic information can be monitored, allowing testing models and assessing their suitability for real-life situations. Additionally, operational information about the received streams can be visually displayed.

The developed monitoring application provides evidence of our implementation of the proposed architecture. The application can communicate with the other architecture's layers through the Communication bus, and consume the resulting enhanced streams, presenting them visually. Finally, it is possible to visualize the extracted information and assess the implemented model's performance at the Semantic Enhancement Layer.

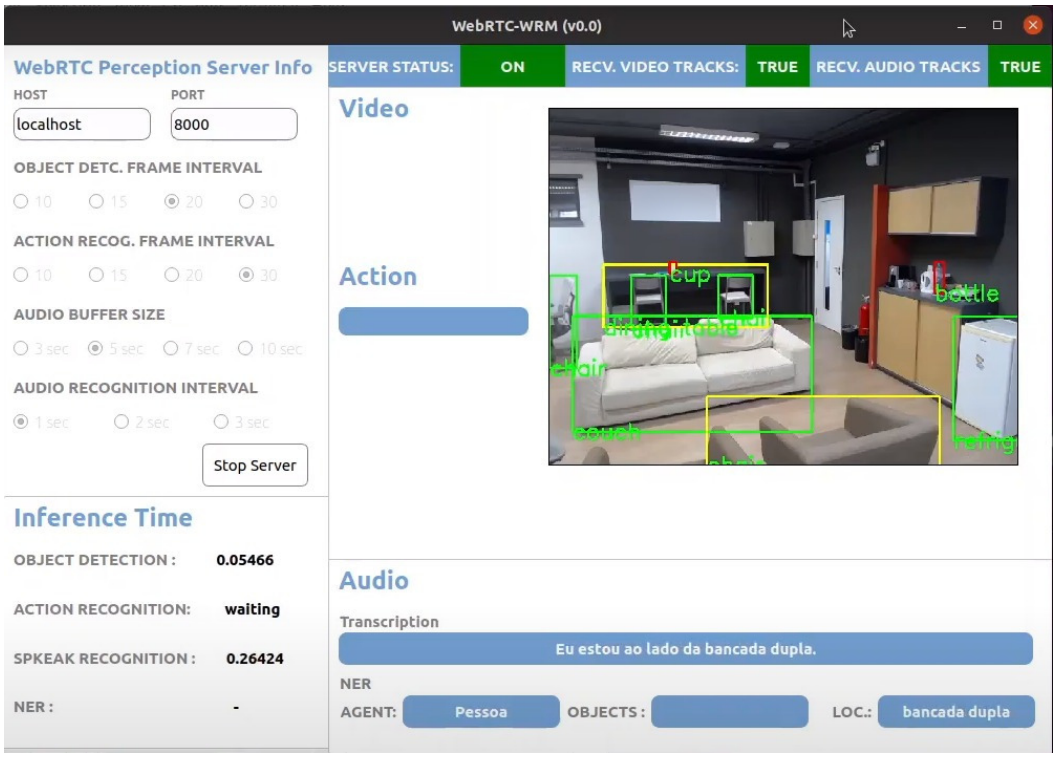


Figure 7.1: Monitor interface during an experiment execution.

7.2 Mobile Robot Perception Module

Once all the architecture layers were implemented, it was tested and integrated with the ASTRO mobile robot simulation. Figure 7.2 illustrate the integration between the simulator application and the other architecture's layers.

To integrate the ASTRO's simulator, a ROS node was developed, which, when starting the library core, also triggers the transmission of the multimedia stream. This includes establishing a connection through the Communication Bus and transmitting the stream via WebRTC. As the enhanced streams are returned to the node, they are published on the "chatter" ROS topic and can be consumed by any module within ROS, in our case, the navigation module. Therefore, the enhanced streams are available for consumption as the robot requires them.

The tests were also carried out in a controlled environment with only a microphone. Through this microphone, a human agent spoke commands and locations that were extracted by the middleware and sent in real time along with the audio stream.

The developed implementation proved applicable to the problem of mobile robots in a collaborative ad hoc environment; the robot was able to navigate through the simulated environment by identifying entities, such as

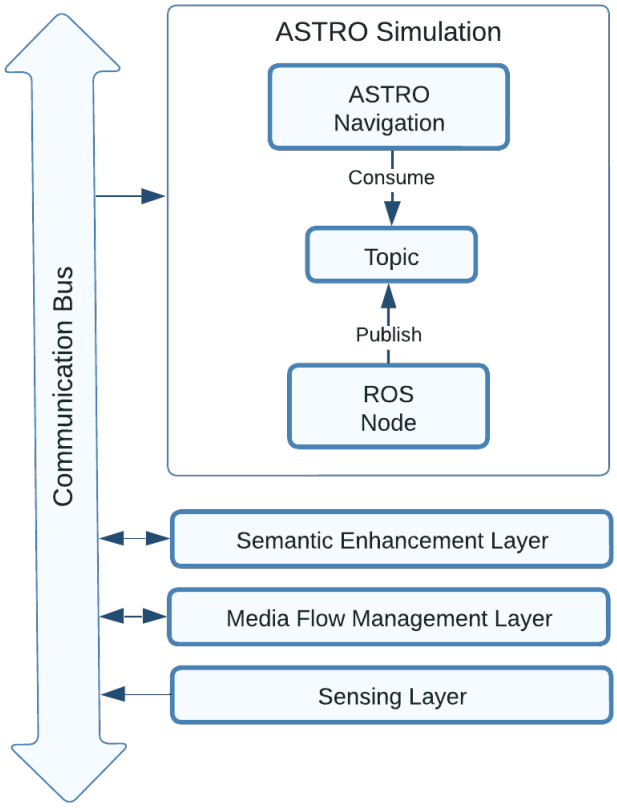


Figure 7.2: Integration of the ASTRO simulation in the architecture.

the location and the agent present in the transcribed text of the captured audio without any need of further processing the stream since it was already enhanced with the relevant information.

8
Conclusion

Traditional multimedia basic services offered to applications are usually limited to supporting the syntactical aspects of efficient representation for storage or for meeting some Quality of Experience (QoE) requirements during reproduction. However, technological trends have brought new challenges to multimedia systems related to exploring a semantic understanding of the content beyond media coding or transmission.

Machine learning technologies have been applied to enrich multimedia content in modern applications, usually without much native support from operating systems, middleware, or development environments. With our present investigation, we argue that basic semantic knowledge of the multimedia content should and can be added to media flows, leveraging more robust development processes of more sophisticated applications by freeing developers of having to implement basic tasks like agent and face detection, action recognition and segmentation, object detection, and classification, among others known machine learning tasks.

To provide those native basic services to applications, our work proposes an architecture for enhancing multimedia flows with semantic information. The proposed architecture consists of four layers and one interconnection bus between them: Sensing, Media Flow Management, Semantic Enhancement, Application, and Communication, respectively. The architecture was implemented and tested in two prominent use cases as a proof of concept for validating functionality and applicability: a monitoring and management application and a mobile robot simulator in a collaborative ad hoc environment.

With the first use-case scenario, the extracted semantic information could be monitored in an experimental setup, allowing testing of the models and assessing their suitability for real-life situations. Operational information about the received streams could be adequately displayed on a monitor, showing an example of how the application communicates and consumes the resulting enhanced streams and presents them visually. Besides visualizing the extracted information, assessing the implemented model's performance at the Semantic Enhancement Layer was also possible.

The second use case involves the Mobile Robot Perception Module simulation, where the enhanced media flows are used for real-world perception and navigation tasks. As this approach provides enhanced information, the robot is equipped with advanced perception capabilities that aid decision-

making processes by improving the robot's understanding of the environment. Furthermore, evaluating the performance of the models implemented by the semantic enhancer helps to improve the analysis of the robustness of the tasks performed by the mobile robot.

One of the main characteristics of the architecture is its versatility in terms of distribution. Clear interfaces allow entities from different layers to be either locally or remotely placed or even distributed across multiple devices according to the application demands. We have shown, in our robot use case, an example in which the process of enriching streams could impose a high cost in terms of algorithmic complexity; hence we opted to delegate this workload to a separate machine, thus saving computational power to keep the robot safe for executing its navigation and decision-making demanded from specific required tasks. Hence, we found that the proposed architecture has shown to be versatile enough to accommodate this distribution while delivering the enriched information as desired.

The architecture presented in this dissertation offers vast potential for application in diverse real-time multimedia semantic processing scenarios. Such scenarios stand to gain significant advantages from this proposed framework, as it enables the utilization of pre-enriched data, eliminating the necessity for internal system resources dedicated to semantic information extraction. Various fields can benefit from this approach, including industry, video monitoring, health, and work safety. The architecture provides a readily accessible repository of high-level information, facilitating the development of robust and efficient solutions in these domains.

8.1 Contribution

As contribution, we proposed a novel middleware architecture for the enrichment of data flows. What sets our approach apart from other related works is its versatility, distribution capability, and potential applicability across a wide range of problems demanding semantic information extraction from data flows.

Compared to related works, the architecture proposed in this dissertation proved to be adaptable to several application contexts, due to the well-defined roles of each layer and the separation of the application logic from the enrichment logic, working in real-time conditions and versatile, being able to have its layers distributed or not.

8.2 Future Works

The next steps include using the enhanced video stream during the decision-making of the mobile robot, helping in its decision-making process. We hope to perform tests on the ASTRO robot to conduct a deeper study of the usability of the proposed architecture in a real scenario. In future works, we also intend to investigate the application of the architecture in other contexts, which could allow us to analyze its effectiveness and performance involving other knowledge enhancement tasks, such as action segmentation and image captioning, for example.

9
Bibliography

AGRAWAL, R. **Multimedia Systems**. A-45, Naraina, Phase-I, New Delhi-110028: EXCEL BOOKS PRIVATE LIMITED, 2013. Produced & Printed by EXCEL BOOKS PRIVATE LIMITED for Lovely Professional University Phagwara. Cited 2 times in pages 21 and 22.

ALGABRI, R.; CHOI, M.-T. Deep-learning-based indoor human following of mobile robot using color feature. **Sensors**, MDPI, v. 20, n. 9, p. 2699, 2020. Cited in page 48.

ALZUBI, J.; NAYYAR, A.; KUMAR, A. Machine learning from theory to algorithms: an overview. In: IOP PUBLISHING. **Journal of physics: conference series**. [S.l.], 2018. v. 1142, p. 012012. Cited 2 times in pages 22 and 23.

BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. **YOLOv4: Optimal Speed and Accuracy of Object Detection**. 2020. Cited in page 25.

BRUNETTI, A. et al. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. **Neurocomputing**, Elsevier, v. 300, p. 17–33, 2018. Cited in page 48.

CARREIRA, J. et al. A short note about kinetics-600. **CoRR**, abs/1808.01340, 2018. Disponível em: <http://arxiv.org/abs/1808.01340>. Cited in page 49.

CARREIRA, J. et al. A short note on the kinetics-700 human action dataset. **CoRR**, abs/1907.06987, 2019. Disponível em: <http://arxiv.org/abs/1907.06987>. Cited in page 49.

CEBOLLADA, S. et al. A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data. **Expert Systems with Applications**, Elsevier, v. 167, p. 114195, 2021. Cited in page 49.

FAN, H. et al. Multiscale vision transformers. **CoRR**, abs/2104.11227, 2021. Disponível em: <https://arxiv.org/abs/2104.11227>. Cited in page 26.

FARIAS, M. Estimating the quality of experience of immersive contents. In: **Proceedings of the 2nd Workshop on Quality of Experience in Visual Multimedia Applications**. New York, NY, USA: Association for Computing Machinery, 2022. (QoEVMA '22), p. 1. ISBN 9781450394994. Disponível em: <https://doi.org/10.1145/3552469.3557784>. Cited in page 14.

FU, H.; QIU, G. Fast semantic image retrieval based on random forest. In: **Proceedings of the 20th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2012. (MM '12), p. 909–912. ISBN 9781450310895. Disponível em: <https://doi.org/10.1145/2393347.2396344>. Cited in page 15.

FURHT, B. Multimedia systems: An overview. **IEEE MultiMedia**, IEEE, v. 1, n. 1, p. 47–59, 1994. Cited 2 times in pages 14 and 22.

GARG, R. et al. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In: LEIBE, B. et al. (Ed.). **Computer Vision – ECCV 2016**. Cham: Springer International Publishing, 2016. p. 740–756. ISBN 978-3-319-46484-8. Cited in page 50.

GIRSHICK, R. **Fast R-CNN**. 2015. Cited in page 25.

GOTZHEIN, R.; GOTZHEIN, R.; WHEELER. **Real-time Communication Protocols for Multi-hop Ad-hoc Networks**. [S.l.]: Springer, 2020. Cited in page 20.

GUPTA, M. et al. A novel vision-based tracking algorithm for a human-following mobile robot. **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, IEEE, v. 47, n. 7, p. 1415–1427, 2016. Cited in page 48.

HANDLEY, M.; JACOBSON, V.; PERKINS, C. **SDP: session description protocol**. [S.l.], 2006. Cited in page 17.

HERCOG, D. **Communication protocols: principles, methods and specifications**. [S.l.]: Springer Nature, 2020. Cited in page 18.

JIANG, P. et al. A review of yolo algorithm developments. **Procedia Computer Science**, v. 199, p. 1066–1073, 2022. ISSN 1877-0509. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 2021): Developing Global Digital Economy after COVID-19. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050922001363>. Cited in page 26.

JIN, Z. et al. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In: **Proceedings of the 25th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2017. (MM '17), p. 795–816. ISBN 9781450349062. Disponível em: <https://doi.org/10.1145/3123266.3123454>. Cited in page 14.

JOCHER, G. et al. **ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation**. Zenodo, 2022. Disponível em: <https://doi.org/10.5281/zenodo.7347926>. Cited in page 25.

KARTHI, M. et al. Evolution of yolo-v5 algorithm for object detection: Automated detection of library books and performace validation of dataset. In: **2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)**. [S.l.: s.n.], 2021. p. 1–6. Cited in page 26.

KAY, W. et al. The kinetics human action video dataset. **CoRR**, abs/1705.06950, 2017. Disponível em: <http://arxiv.org/abs/1705.06950>. Cited in page 49.

KONSTANTINOU, N. et al. A context-aware middleware for real-time semantic enrichment of distributed multimedia metadata. **Multimedia Tools and Applications**, Springer, v. 46, p. 425–461, 2010. Cited 4 times in pages 14, 30, 32, and 33.

KOVÁCS, Z. L. **Redes neurais artificiais**. São Paulo, São Paulo: Editora Livraria da Fisica, 2002. Cited in page 23.

LEW, M. S. et al. Content-based multimedia information retrieval: State of the art and challenges. **ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)**, ACM New York, NY, USA, v. 2, n. 1, p. 1–19, 2006. Cited in page 14.

LI, J. et al. A survey on deep learning for named entity recognition. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 34, n. 1, p. 50–70, 2020. Cited 2 times in pages 50 and 51.

LI, Y. et al. Mvitv2: Improved multiscale vision transformers for classification and detection. **2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, p. 4794–4804, 2021. Cited 3 times in pages 26, 27, and 49.

LIDON, A. et al. Semantic summarization of egocentric photo stream events. In: **Proceedings of the 2nd Workshop on Lifelogging Tools and Applications**. New York, NY, USA: Association for Computing Machinery, 2017. (LTA '17), p. 3–11. ISBN 9781450355032. Disponível em: <https://doi.org/10.1145/3133202.3133204>. Cited in page 15.

LIN, T.-Y. et al. Microsoft coco: Common objects in context. In: FLEET, D. et al. (Ed.). **Computer Vision – ECCV 2014**. Cham: Springer International Publishing, 2014. p. 740–755. ISBN 978-3-319-10602-1. Cited in page 50.

LORETO, S.; ROMANO, S. P. Real-time communications in the web: Issues, achievements, and ongoing standardization efforts. **IEEE Internet Computing**, IEEE, v. 16, n. 5, p. 68–73, 2012. Cited in page 46.

MACENSKI, S. et al. Robot operating system 2: Design, architecture, and uses in the wild. **Science Robotics**, v. 7, n. 66, p. eabm6074, 2022. Disponível em: <https://www.science.org/doi/abs/10.1126/scirobotics.abm6074>. Cited in page 43.

MCNAMEE, P.; MAYFIELD, J. Entity extraction without language-specific resources. In: **Proceedings of the 6th Conference on Natural Language Learning - Volume 20**. USA: Association for Computational Linguistics, 2002. (COLING-02), p. 1–4. Disponível em: <https://doi.org/10.3115/1118853.1118873>. Cited in page 51.

MELO, F. S.; SARDINHA, A. Ad hoc teamwork by learning teammates' task. **Autonomous Agents and Multi-Agent Systems**, Springer, v. 30, p. 175–219, 2016. Cited 3 times in pages 21, 42, and 49.

MELO, F. S. et al. Project inside: towards autonomous semi-unstructured human–robot social interaction in autism therapy. **Artificial intelligence in medicine**, Elsevier, v. 96, p. 198–216, 2019. Cited in page 43.

MENG, K. et al. A deep multi-modal fusion approach for semantic place prediction in social media. In: **Proceedings of the Workshop on Multimodal Understanding of Social, Affective and Subjective Attributes**. New York, NY, USA: Association for Computing Machinery, 2017. (MUSA2 '17), p. 31–37. ISBN 9781450355094. Disponível em: <https://doi.org/10.1145/3132515.3132519>. Cited in page 15.

MIRSKY, R. et al. A survey of ad hoc teamwork research. In: BAUMEISTER, D.; ROTHE, J. (Ed.). **Multi-Agent Systems - 19th European Conference, EUMAS 2022, Düsseldorf, Germany, September 14-16, 2022, Proceedings**. Berlin, Heidelberg: Springer, 2022. (Lecture Notes in Computer Science, v. 13442), p. 275–293. Disponível em: <https://doi.org/10.1007/978-3-031-20614-6_16>. Cited 3 times in pages 20, 21, and 42.

MOBASHER, B.; COOLEY, R.; SRIVASTAVA, J. Automatic personalization based on web usage mining. **Communications of the ACM**, ACM New York, NY, USA, v. 43, n. 8, p. 142–151, 2000. Cited in page 14.

MONTAGNUOLO, M. et al. Real time semantic enrichment of broadcast content in the big data age. In: **2017 IEEE International Conference on Big Data (Big Data)**. Boston, MA: IEEE, 2017. p. 1704–1708. Cited 2 times in pages 30 and 33.

MORIKAWA, C.; SILVA, G. C. de. User interaction techniques for multimedia retrieval. In: **Proceedings of the 2012 Joint International Conference on Human-Centered Computer Environments**. New York, NY, USA: Association for Computing Machinery, 2012. (HCCE '12), p. 68–75. ISBN 9781450311915. Disponível em: <https://doi.org/10.1145/2160749.2160765>. Cited in page 14.

NADEAU, D.; TURNEY, P. D.; MATWIN, S. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In: LAMONTAGNE, L.; MARCHAND, M. (Ed.). **Advances in Artificial Intelligence**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 266–277. ISBN 978-3-540-34630-2. Cited in page 51.

NAHRSTEDT, K.; BALKE, W.-T. Towards building large scale multimedia systems and applications: challenges and status. In: **Proceedings of the first ACM international workshop on Multimedia service composition**. [S.l.: s.n.], 2005. p. 3–10. Cited in page 14.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: **Icml**. New York, New York: ACM, 2010. Cited in page 24.

NECULA, S.-C. et al. Enhancement of e-commerce websites with semantic web technologies. **Sustainability**, MDPI, v. 10, n. 6, p. 1955, 2018. Cited in page 14.

NEVES, A.; SARDINHA, A. Learning to cooperate with completely unknown teammates. **arXiv preprint arXiv:2205.03289**, 2022. Cited in page 20.

NIELSEN, M. A. **Neural networks and deep learning**. San Francisco, CA: Determination press, 2015. v. 25. Cited 2 times in pages 23 and 24.

OLATUNJI, I. E. Human activity recognition for mobile robot. **Journal of Physics: Conference Series**, IOP Publishing, v. 1069, n. 1, p. 012148, aug 2018. Disponível em: <https://dx.doi.org/10.1088/1742-6596/1069/1/012148>. Cited in page 49.

QUIGLEY, M. et al. Ros: an open-source robot operating system. In: **ICRA workshop on open source software**. Kobe, Japan: IEEE, 2009. v. 3, p. 5. Cited 2 times in pages 28 and 29.

RADFORD, A. et al. **Robust Speech Recognition via Large-Scale Weak Supervision**. 2022. Cited 3 times in pages 27, 28, and 50.

RAHAMAN, M. H. A survey on real-time communication for web. **Scientific Res. J**, v. 3, n. 7, p. 39–45, 2015. Cited in page 46.

REDMON, J. et al. **You Only Look Once: Unified, Real-Time Object Detection**. 2016. Cited in page 25.

REDMON, J. et al. You only look once: Unified, real-time object detection. In: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Los Alamitos, CA, USA: IEEE Computer Society, 2016. p. 779–788. ISSN 1063-6919. Disponível em: <https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.91>. Cited in page 50.

REDMON, J.; FARHADI, A. **YOLO9000: Better, Faster, Stronger**. 2016. Cited in page 25.

REDMON, J.; FARHADI, A. **YOLOv3: An Incremental Improvement**. 2018. Cited in page 25.

RIBEIRO, J. G. et al. HOTSPOT: An Ad Hoc Teamwork Platform for Mixed Human-Robot Teams. 2021. Disponível em: <https://www.techrxiv.org/articles/preprint/HOTSPOT_An_Ad_Hoc_Teamwork_Platform_for_Mixed_Human-Robot_Teams/17026013>. Cited 2 times in pages 50 and 51.

SALVATORE, L. **Real-Time Communication with WebRTC**. [S.l.]: O'Reilly, 2014. Cited in page 19.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of research and development**, IBM, v. 3, n. 3, p. 210–229, 1959. Cited in page 22.

SANABRIA, M. et al. A deep architecture for multimodal summarization of soccer games. In: **Proceedings Proceedings of the 2nd International Workshop on Multimedia Content Analysis in Sports**. New York, NY, USA: Association for Computing Machinery, 2019. (MMSports '19), p. 16–24. ISBN 9781450369114. Disponível em: <https://doi.org/10.1145/3347318.3355524>. Cited in page 14.

SCHLEGEL, C. et al. Vision based person tracking with a mobile robot. In: CITESEER. **BMVC**. Pennsylvania: Citeseer, 1998. p. 1–10. Cited in page 48.

SEKINE, S.; NOBATA, C. Definition, dictionaries and tagger for extended named entity hierarchy. In: **Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)**. Lisbon, Portugal: European Language Resources Association (ELRA), 2004. Disponível em: <http://www.lrec-conf.org/proceedings/lrec2004/pdf/65.pdf>. Cited in page 51.

SHANG, X. et al. Multimodal video summarization via time-aware transformers. In: **Proceedings of the 29th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2021. (MM '21), p. 1756–1765. ISBN 9781450386517. Disponível em: <https://doi.org/10.1145/3474085.3475321>. Cited in page 14.

SOARES, E. R.; BARRéRE, E. A framework for automatic topic segmentation in video lectures. In: **Anais Estendidos do XXIV Simpósio Brasileiro de Sistemas Multimídia e Web**. Porto Alegre, RS, Brasil: SBC, 2018. p. 31–36. ISSN 2596-1683. Disponível em: <https://sol.sbc.org.br/index.php/webmedia_estendido/article/view/4053>. Cited 2 times in pages 31 and 33.

SREDOJEV, B.; SAMARDZIJA, D.; POSARAC, D. Webrtc technology overview and signaling solution design and implementation. In: IEEE. **2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)**. [S.l.], 2015. p. 1006–1009. Cited 2 times in pages 18 and 19.

SRISURESH, P.; EGEVANG, K. **Traditional IP network address translator (Traditional NAT)**. [S.l.], 2001. Cited in page 18.

STONE, P. et al. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In: **Proceedings of the Twenty-Fourth Conference on Artificial Intelligence**. [S.l.: s.n.], 2010. Cited 2 times in pages 16 and 20.

ŠVEC, J.; NEDUCHAL, P.; HRÚZ, M. Multi-modal communication system for mobile robot. **IFAC-PapersOnLine**, Elsevier, v. 55, n. 4, p. 133–138, 2022. Cited 2 times in pages 31 and 33.

SZARVAS, G.; FARKAS, R.; KOCSOR, A. A multilingual named entity recognition system using boosting and c4.5 decision tree learning algorithms. In: TODOROVSKI, L.; LAVRAČ, N.; JANTKE, K. P. (Ed.). **Discovery Science**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 267–278. ISBN 978-3-540-46493-8. Cited in page 51.

WAN, J. et al. Deep learning for content-based image retrieval: A comprehensive study. In: **Proceedings of the 22nd ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2014. (MM '14), p. 157–166. ISBN 9781450330633. Disponível em: <https://doi.org/10.1145/2647868.2654948>. Cited in page 14.

WANG, X.; WANG, Y. Improving content-based and hybrid music recommendation using deep learning. In: **Proceedings of the 22nd ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2014. (MM '14), p. 627–636. ISBN 9781450330633. Disponível em: <https://doi.org/10.1145/2647868.2654940>. Cited in page 14.

WANG, Y. et al. **InternVideo: General Video Foundation Models via Generative and Discriminative Learning**. 2022. Cited in page 49.

WEI, Y. et al. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In: **Proceedings of the 27th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2019. (MM '19), p. 1437–1445. ISBN 9781450368896. Disponível em: <https://doi.org/10.1145/3343031.3351034>. Cited in page 14.

YU, D.; DENG, L. **Automatic speech recognition**. London. UK: Springer, 2016. v. 1. Cited in page 50.

ZHANG, D. et al. Smart multi-modal marine monitoring via visual analysis and data fusion. In: **Proceedings of the 2nd ACM International Workshop on Multimedia Analysis for Ecological Data**. New York, NY, USA: Association for Computing Machinery, 2013. (MAED '13), p. 29–34. ISBN 9781450324014. Disponível em: <https://doi.org/10.1145/2509896.2509903>. Cited in page 14.

ZHANG, K. et al. Joint face detection and alignment using multitask cascaded convolutional networks. **IEEE Signal Processing Letters**, v. 23, n. 10, p. 1499–1503, Oct 2016. ISSN 1070-9908. Cited in page 49.

ZHAO, Z.-Q. et al. Object detection with deep learning: A review. **IEEE transactions on neural networks and learning systems**, IEEE, v. 30, n. 11, p. 3212–3232, 2019. Cited in page 50.