



Escuela
Politécnica
Superior

Indoor Risks Assessment Using Video Captioning



Bachelor's degree in Computer
Engineering

Bachelor's thesis

Author:

Javier Rodríguez Juan

Supervisors:

Jose García Rodríguez

David Tomás Díaz

May 2023



Universitat d'Alacant
Universidad de Alicante

Indoor Risks Assessment Using Video Captioning

Author

Javier Rodríguez Juan

Supervisors

Jose García Rodríguez

Departamento de Tecnología Informática y Computación

David Tomás Díaz

Departamento de Lenguajes y Sistemas Informáticos



Bachelor's degree in Computer Engineering



Escuela
Politécnica
Superior



Universitat d'Alacant
Universidad de Alicante

ALICANTE, May 2023

Abstract

The progress of automatic scene analysis techniques for homes and the development of ambient assisted living systems is vital to help different kinds of people, such as the elderly or visually impaired individuals, who require special care in their daily lives. In this bachelor's thesis we are going to develop a study of the most promising used techniques inside the Video Captioning and scene analysis scope and we will propose a Deep Learning pipeline aimed at performing Risks Assessment on input videos using the knowledge acquired during the study. This can be potentially applied to create systems aimed to help aforementioned people. Moreover, we will propose different evaluation architectures to test each of the stages involved in the Risks Assessment pipeline in order to observe its effectiveness and limitations.

In this work we will introduce SwinBERT, a powerful and recent Video Captioning model, complemented with YOLOv7, a model aimed at the Object Recognition task, for the analysis of home scenes. Moreover, we will use various lexical transformations and linguistic models to maximize the semantic similarity of descriptions generated and objects detected, aligning them with the annotations provided by the datasets used. This approach will allow us to achieve more accurate matches from a human perspective. In the experiments we will outstand the usage of the large-scale dataset Charades, which was created with the goal of producing a vast dataset designed for the visual analysis, while preserving the naturalness and spontaneity of household and daily activities.

Resumen

El avance de las técnicas de análisis automático de escenas para hogares y el desarrollo de sistemas de asistencia en residencias es vital para ayudar a diferentes tipos de personas, como ancianos o individuos con discapacidades visuales, que requieren cuidados especiales en su vida diaria. En este proyecto vamos a desarrollar un estudio de las técnicas más prometedoras utilizadas dentro del ámbito de la descripción de vídeos y del análisis de escenas para finalmente proponer una arquitectura, basada en Deep Learning, destinada a realizar una evaluación de riesgos sobre un vídeo de entrada utilizando el conocimiento adquirido durante el estudio. Esto puede ser potencialmente aplicado para crear sistemas orientados a ayudar a las personas anteriormente mencionadas. Además, propondremos diferentes arquitecturas destinadas a la evaluación de cada una de las etapas involucradas en la arquitectura de evaluación de riesgos con el fin de observar su efectividad y limitaciones.

En este trabajo introduciremos SwinBERT, un potente y reciente modelo de descripción de videos, complementado con YOLOv7, un modelo orientado a la tarea de reconocimiento de objetos, para el análisis de escenas domésticas. Además, utilizaremos varias transformaciones léxicas y modelos lingüísticos para maximizar la similitud semántica de las descripciones generadas y objetos detectados, alineándolos con las anotaciones proporcionadas por los conjuntos de datos utilizados. Este enfoque nos permitirá lograr coincidencias más precisas desde una perspectiva humana. En los experimentos destacaremos el uso del conjunto de datos a gran escala Charades, que se creó con el objetivo de producir un amplio conjunto de datos diseñado para el análisis visual, preservando al mismo tiempo la naturalidad y espontaneidad de las actividades domésticas y cotidianas.

Acknowledgements

I would like to thank the efforts, help and guidance given by the supervisors of this thesis, Jose García Rodríguez and David Tomás Díaz. Also I would also like to extend my thanks to the 3D Perception Lab for their constant availability and support whenever I needed assistance. Finally, I would like to give a special thanks to my family and close friends for their unwavering encouragement and motivation throughout my Bachelor's Degree.

*When you want something intensely enough,
no sacrifice is too great*

Rafael Nadal.

Contents

1. Introduction	1
1.1. Overview	1
1.2. Motivation	1
1.3. Proposal and goals	2
1.4. Timeline	2
1.4.1. Learning Stage	3
1.4.2. Research Stage	3
1.4.3. Implementation Stage	4
1.4.4. Experimentation Stage	4
1.5. Outline	5
2. State of the art	7
2.1. Neural Networks	7
2.1.1. Convolutional Neural Networks	7
2.1.2. Recurrent Neural Networks	8
2.1.3. Transformers	9
2.2. Captioning	11
2.2.1. Image Captioning	11
2.2.2. Video Captioning	11
2.3. Video Captioning in depth	13
2.3.1. Fundamental Structure	14
2.3.2. Encoding stage (CNN)	15
2.3.3. Decoding stage (RNN)	16
2.3.4. Attention models	16
2.3.5. Indoor scene captioning	17
2.4. Dementia	18
2.5. Dementia and AAL datasets	19
2.5.1. DementiaBank	20
2.5.2. Dem@Care	20
2.5.3. Charades	21
2.5.4. ETRI-Activity3D	22
2.5.5. VirtualHome	23
3. Methodology	25
3.1. Hardware	25
3.2. Software	25
3.3. Machine learning frameworks	26
3.3.1. TensorFlow	27
3.3.2. PyTorch	27

3.3.3. HuggingFace Transformers	27
3.3.4. OpenCV	28
3.3.5. SpaCy	28
3.3.6. Gensim	28
3.4. Selected technologies	29
4. Proposed architecture	31
4.1. Datasets	31
4.1.1. Requirements	31
4.1.2. Selection	31
4.1.3. Risks-objects corpus	33
4.2. Modules	34
4.2.1. Video captioning module	34
4.2.2. Pre-processing module	36
4.2.3. Object extraction module	36
4.2.4. Object recognition module	37
4.2.5. Risks matching module	39
4.3. Evaluation	40
4.3.1. BLEU Score	40
4.3.2. BERT Score	41
4.3.3. Zero-shot classification	41
4.3.4. Object Recognition Recall Score	42
5. Experiments and results	45
5.1. Descriptions evaluation	45
5.1.1. Zero-shot classification-based pipeline	45
5.1.2. Metric-based evaluation pipeline	48
5.2. Object Recognition evaluation	52
5.2.1. Description	53
5.2.2. Results	53
5.3. Risks Assessment	56
5.3.1. Description	56
5.3.2. Results	57
6. Conclusions	63
6.1. Conclusions	63
6.2. Future works	64
6.3. Publications	65
Bibliography	67
List of Acronyms and Abbreviations	73
A. Appendix I	75

B. Appendix II	77
B.1. Objects corpus	77
B.2. Properties corpus	84
C. Appendix III	89

List of Figures

1.1.	Gantt diagram of tasks developed during the entire thesis.	3
1.2.	Tasks and associated temporal development periods in the learning stage. . .	3
1.3.	Tasks and associated temporal development periods in the research stage. . .	4
1.4.	Tasks and associated temporal development periods in the implementation stage.	4
1.5.	Tasks and associated temporal development periods in the experimentation stage.	5
2.1.	Basic Artificial Neural Network structure. Retrieved from Kumar et al. (2020).	8
2.2.	Basic intuition of Convolutional Neural Network (CNN) structure. Retrieved from Timilsina et al. (2019).	9
2.3.	Basic intuition of Recurrent Neural Network (RNN) structure. Retrieved from Villamizar Torres & Lizarazo (2019).	10
2.4.	Basic intuition of Transformers structure. Retrieved from Mishev et al. (2020).	10
2.5.	Examples of image captioning descriptions.	12
2.6.	Difference between Single sentence Video Captioning and Dense Video Captioning. Retrieved from Islam et al. (2021).	13
2.7.	Resume of outlined Video Captioning structures. Retrieved from Jain et al. (2022).	14
2.8.	Example of overlapping bounding boxes removals. Retrieved from https://stanford.edu/\protect\unhbox\voidb@x\protect\penalty\@M\{}shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks	15
2.9.	Example of possible Video Captioning framework. Retrieved from Jain et al. (2022).	16
2.10.	Attention applied to generate a video frame caption. Retrieved from https://stanford.edu/\protect\unhbox\voidb@x\protect\penalty\@M\{}shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks	17
2.11.	Average percentage of the different types of dementia.	18
2.12.	Map of VR test for dementia patients. Retrieved from Puthusserypady et al. (2022).	19
2.13.	This picture is called the Cookie Theft, it is a common test image used in picture description tasks.	21
2.14.	Example of test environment for participants of Dem@Care project.	22
2.15.	Charades dataset sample frame.	22
2.16.	ETRI dataset sample frames.	23
2.17.	Simulation frames and descriptions from the ActivityPrograms and Virtual-Home Activity datasets.	24
3.1.	Sample command to run a container through the docker CLI.	26

3.2. Popularity comparison of TensorFlow and Pytorch inside the research community.	30
4.1. Example of object included in the Risks-objects corpus. The cup object will acquire the risk <i>breakable</i> associated with the property <i>fragile</i> and the ones associated with the property <i>liquid_container</i> . Objects can have no specific risks.	34
4.2. Example of property included in the Risks-objects corpus.	34
4.3. SwinBERT internal architecture.	35
4.4. Applying individual transformation over a sample sentence.	36
4.5. Execution sample of the object extraction module over Charades input video.	38
4.6. Execution sample of the YOLOv7 model over a Charades input video.	39
4.7. Comparison between variants of BLEU score.	41
4.8. Example of BERT Score evaluation using F1. The candidate with highest semantic similarity achieves the highest score.	42
4.9. Example of Zero-shot classification determining scores of a sentence given 3 possible labels.	42
4.10. Recall formula.	43
5.1. Zero-shot classification of SwinBERT descriptions experiment pipeline.	47
5.2. Sample of wrong labeling given a correct description.	48
5.3. Sample of wrong labeling because of unexpected description.	49
5.4. BLEU-1 and BERT Score evaluation of SwinBERT descriptions experiment pipeline.	50
5.5. Charades' frame example with its ground-truth (GT) annotation and three different captions obtained from SwinBERT (MSVD, MSR-VTT and VATEX).	51
5.6. Pipeline used for computing the evaluation of the Object Recognition proposed architecture. COCO classes are extracted from the Risks-objects corpus as this corpus is based on our subset of COCO's 56 classes.	54
5.7. Objects extracted from YOLO and different SwinBERT pretrainings over an input Charades video. Observe confusion recognising the screen as a laptop.	55
5.8. Pipeline used to compute Risks Assessment.	58
5.9. Risks Assessment over a Charades dataset input video. Although the pipeline momentarily confuses the TV (screen computer) with a laptop all the risks detected are still valid.	59
5.10. Detections associated to the Figure 5.9. In red the TV/Laptop, in orange the chairs, in green the cellphone, in cyan blue the remote, in dark blue the cup, in purple the book and in magenta the couch.	60
5.11. Detections associated to the Figure 5.12. In cyan blue the bottle, in green the refrigerator	60
5.12. Risks Assessment over an ETRI-Activity3D input video.	61
B.1. Fragment of the objects file which comprises the Risks-objects corpus (Object 1-7).	77
B.2. Fragment of the objects file which comprises the Risks-objects corpus (Object 8-15).	78

B.3. Fragment of the objects file which comprises the Risks-objects corpus (Object 16-24).	79
B.4. Fragment of the objects file which comprises the Risks-objects corpus (Object 24-32).	80
B.5. Fragment of the objects file which comprises the Risks-objects corpus (Object 33-40).	81
B.6. Fragment of the objects file which comprises the Risks-objects corpus (Object 41-48).	82
B.7. Fragment of the objects file which comprises the Risks-objects corpus (Object 49-56).	83
B.8. Fragment of the properties file which comprises the Risks-objects corpus (Property 1-4).	84
B.9. Fragment of the properties file which comprises the Risks-objects corpus (Property 5-7).	85
B.10. Fragment of the properties file which comprises the Risks-objects corpus (Property 8-11).	86
B.11. Fragment of the properties file which comprises the Risks-objects corpus (Property 12-14).	87
C.1. This figure represents a demonstration of the low amount of object annotations provided by Charades. Although a rich set of objects is detected, none of these are in the annotations set so the Recall score for the Object Recognition task in this video is 0,0.	89
C.2. This figure represents a demonstration of how poor quality videos affects Objects Recognition results. This frame shows a person drinking a cup of coffee. As the video has poor quality both SwinBERT and YOLOv7 are not able to recognize any object (only recognizes the person). The annotations objects set of this video is only composed by the object "Cup".	90

List of Tables

3.1. Specification details of Asimov.	26
4.1. Summary table of data provided by Charades and ETRI-Activity3D datasets.	32
4.2. Complete set of 56 objects that compose our Risks-objects corpus used during experimentation.	33
5.1. BLEU-1 score results of testing a sample of 3,000 videos with different textual preprocessing methods.	51
5.2. BERT Score results (precision and recall) of testing a sample of 3,000 videos with different textual preprocessings.	52
5.3. Recall evaluation of the Object Recognition pipeline over 1430 subset of videos from Charades dataset. Last row represents Recall when only using YOLO detections.	56
A.1. Recall percentage obtained from testing Zero-shot classification for each of the ETRI-Activity3D dataset's actions (From action 1-34).	75
A.2. Recall percentage obtained from testing Zero-shot classification for each of the ETRI-Activity3D dataset's actions (From action 35-55).	76

1. Introduction

In this chapter, we will introduce the various motivations and goals that have led to the development of this bachelor's thesis. Additionally, we will present a timeline outlining the different development phases of this work. In Section 1.1 we provide a synopsis of the entire document. Following that, in Section 1.2 we discuss the different motivations that have driven the development of this thesis. Subsequently, in Section 1.3, we outline a list of goals proposed to be achieved during this work. Moving on, Section 1.4 provides an organized overview of the various development phases and tasks performed. Finally, in Section 1.5, we mention and explain the remaining chapters that comprise this work.

1.1. Overview

This thesis proposes a framework for assessing risks in indoor scene videos using Video Captioning and Object Recognition models. Additionally, it presents various deep learning-based architectures for analyzing indoor scene videos, with applications in fields such as robotics, surveillance systems, and Ambient Assisted Living (AAL) for individuals with special needs, including the elderly and visually impaired.

The proposed architectures are built upon previous research conducted on state-of-the-art Video Captioning and evaluation techniques. To achieve this, we will review the current most outstanding Video Captioning models and explore different evaluation methods applicable to their results. Our primary focus will be on studying different frameworks for home Video Captioning, utilizing SwinBERT, a recent and promising Video Captioning model. We will enhance the descriptions generated by SwinBERT using various linguistic processing techniques to maximize the semantic information present in the outputs. Furthermore, we will complement the SwinBERT descriptions by incorporating an Object Recognition model such as YOLOv7, which will improve the detection rate of objects. Additionally, we will review datasets with various types of features.

1.2. Motivation

Nowadays, many people suffer from diseases and impairments that prevent them from leading independent and ordinary lives. One example of this is dementia, a condition that affects a significant portion of our society, causing a progressive decline in cognitive functions and hindering individuals from performing daily tasks as the disease progresses.

Visual analysis systems have seen rapid advancements in performance, and it has been proven that robust and powerful systems can be built using Computer Vision techniques. It is widely believed that these technologies can be leveraged to create assistive systems that improve the quality of life for dependent individuals. The intensive development of such

studies is expected to make a significant difference in their lives in the coming years, and this is one of the main motivations behind this thesis.

Another motivation for undertaking this thesis was the collaboration with the Languages and Computer Systems Department (Departamento de Lenguajes y Sistemas Informáticos (DLSI) in Spanish) and the support from the Computation and IT Department (Departamento de Tecnología Informática y Computación (DTIC) in Spanish), particularly the assistance provided by the 3D Perception Lab group. This group specializes in GPU computing, 3D computation, and Computer Vision, and they are involved in the development of the national project Monitoring and Detection of human behaviors for personalized assistance and early disease detection, A2HUMPA in Spanish (MoDeAss), led by researchers José García Rodríguez and Miguel Angel Cazorla-Quevedo. The MoDeAss project focuses on the analysis of human behaviors for monitoring, personalized assistance, and early disease detection, primarily aimed at helping dependent individuals in their homes. This thesis is also driven by the desire to contribute to this project.

On a personal level, throughout my Bachelor's Degree, I have always been curious about how artificial intelligence operates behind the scenes and how these systems can make decisions autonomously, similar to humans. This curiosity led me to conduct independent research in this field and I was amazed by the incredible work that has been accomplished over the years. Inspired by these achievements, I wanted to delve deeper and pursue a career in artificial intelligence. Therefore, this thesis represents the first step in my personal journey within the field of artificial intelligence.

1.3. Proposal and goals

In this thesis, our primary objective is to develop a visual analysis framework for processing home environment videos that generates a list of potential risks based on the recognized objects in the scene. To accomplish this, we will refer to a well-studied Deep Learning (DL) task known as Video Captioning, which focuses on generating natural language descriptions from input videos. Additionally, we will conduct a comprehensive review of state-of-the-art models, techniques, and datasets employed in the field of Video Captioning. Using this knowledge, we will perform experiments on various visual analysis architectures to demonstrate the effectiveness and limitations of the reviewed resources in Computer Vision analysis for home scenes. This represents our second goal. Since our thesis involves generating descriptions from videos, it entails not only Computer Vision tasks but also Natural Language Processing (NLP) tasks. Hence, our third goal is to provide insights into how linguistic processing techniques and lexical transformations applied to the generated descriptions can impact evaluation results. To fulfill this objective, we will leverage popular NLP-oriented libraries available today.

1.4. Timeline

In this section, we will introduce the different stages involved in the development of this thesis. We will provide a chronological timeline in which the core tasks performed during this work are mentioned and ordered temporally. The development of the thesis comprised four main stages: the learning stage, the research stage, the implementation stage, and finally,

the experimentation stage. The composition of this thesis commenced in September 2022 and concluded in May 2023, encompassing a total of 9 months of development. In Figure 1.1 a Gantt diagram which illustrates all different tasks developed during the different stages of the thesis is presented.



Figure 1.1: Gantt diagram of tasks developed during the entire thesis.

1.4.1. Learning Stage

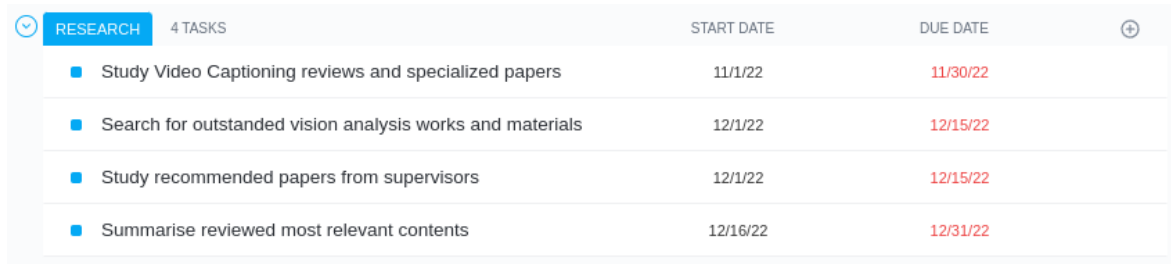
The first stage was characterized by completing machine learning courses and tutorials to acquire the fundamentals and basics of machine and deep learning. During this stage, we not only completed theoretical courses but also conducted practical tutorials and created initial experimental scripts to facilitate the learning curve of frameworks and libraries related to machine learning. This stage took place between September and October.

LEARNING	2 TASKS	START DATE	DUE DATE
■	Completing machine learning courses	9/1/22	9/30/22
■	Review HuggingFace and Pytorch tutorials	10/1/22	10/31/22

Figure 1.2: Tasks and associated temporal development periods in the learning stage.

1.4.2. Research Stage

In the second stage, we delved into Video Captioning and Computer Vision analysis, reading surveys and specialized papers to explore the state-of-the-art in these specialties. During this stage, supervisors also recommended interesting papers that were considered for inclusion in the state-of-the-art chapter of the thesis. As a final step of this stage, a summary of the most relevant findings was created, and the state-of-the-art chapter of this thesis was started, using the summary of materials, methods, and investigated models as a reference. This stage took place between November and December.

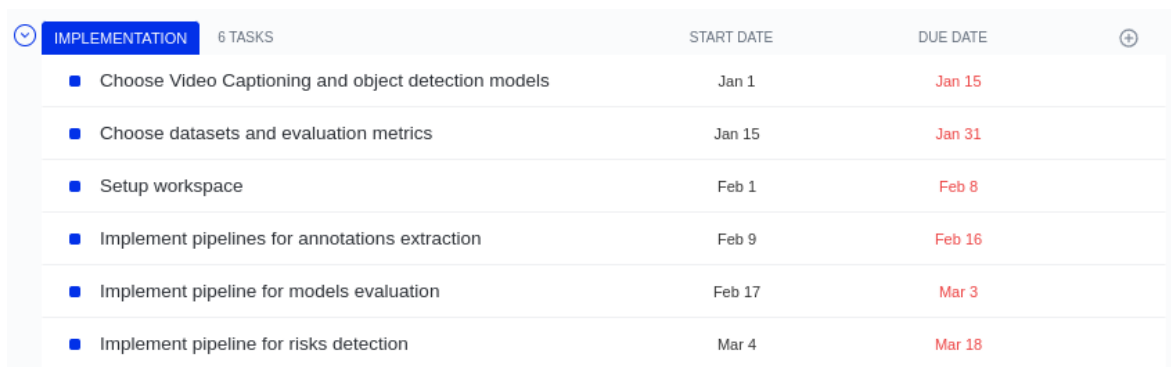


RESEARCH	4 TASKS	START DATE	DUE DATE
■	Study Video Captioning reviews and specialized papers	11/1/22	11/30/22
■	Search for outstanded vision analysis works and materials	12/1/22	12/15/22
■	Study recommended papers from supervisors	12/1/22	12/15/22
■	Summarise reviewed most relevant contents	12/16/22	12/31/22

Figure 1.3: Tasks and associated temporal development periods in the research stage.

1.4.3. Implementation Stage

In the third stage, we made decisions regarding the materials, such as models and datasets, to be used for implementing the different pipelines needed. After making these decisions, we focused our efforts on setting up the environment to ensure the correct execution of the selected materials. Finally, we constructed the proposed architectures and implemented the pipelines to test and evaluate them using the selected technologies and frameworks. This stage took place between January and February.

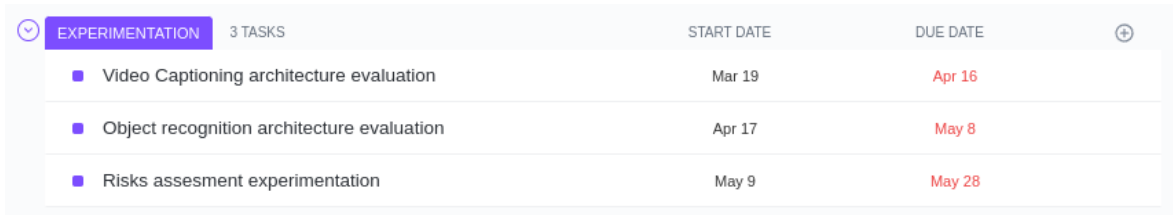


IMPLEMENTATION	6 TASKS	START DATE	DUE DATE
■	Choose Video Captioning and object detection models	Jan 1	Jan 15
■	Choose datasets and evaluation metrics	Jan 15	Jan 31
■	Setup workspace	Feb 1	Feb 8
■	Implement pipelines for annotations extraction	Feb 9	Feb 16
■	Implement pipeline for models evaluation	Feb 17	Mar 3
■	Implement pipeline for risks detection	Mar 4	Mar 18

Figure 1.4: Tasks and associated temporal development periods in the implementation stage.

1.4.4. Experimentation Stage

In the final stage of development, we utilized the pipelines developed during the implementation stage to evaluate the proposed architectures and draw conclusions from the results. This stage involved two main tasks: performance evaluation of the selected models using the chosen datasets and evaluation of the Risks Assessment pipeline. These tasks were performed iteratively, involving three phases: testing, analysis, and modifications. During the testing phase, we obtained results from the proposed architectures. In the analysis phase, we examined the obtained results, and in the modifications phase, we reflected on the results to identify areas for improvement and propose changes that could enhance the results. This iterative approach allowed us to test the architectures from different perspectives and establish future work to be carried out. This stage took place between March and May.



EXPERIMENTATION	3 TASKS	START DATE	DUE DATE
■	Video Captioning architecture evaluation	Mar 19	Apr 16
■	Object recognition architecture evaluation	Apr 17	May 8
■	Risks assesment experimentation	May 9	May 28

Figure 1.5: Tasks and associated temporal development periods in the experimentation stage.

1.5. Outline

This thesis consists of six core chapters. The rest of the document is organised as follows: In Chapter 2, we review the current techniques, models, and datasets in Video Captioning that are necessary for the development of the rest of the thesis. Afterward, in Chapter 3, we present the infrastructure and technologies used for the experimentation. In Chapter 4, we introduce the different components that will be used as references. Then, in Chapter 5, we present the proposed architectures, including the evaluation results. Finally, in Chapter 6, we summarize all the contributions and present future work, concluding the thesis.

2. State of the art

In this second chapter, we will present the state-of-the-art in Video Captioning and provide an introduction to dementia and AAL scopes. The purpose of this section is to present the most significant works and discoveries in these areas, aiming to familiarize the reader with the experimental context that will be explored in subsequent sections. This chapter is organized as follows: Section 2.1 provides an overview of the fundamental concepts of neural networks, offering the reader an intuition of their principles. Subsequently, in Section 2.2, we delve into the origins of the Video Captioning task. Following that, Section 2.3 explores the depth of the Video Captioning scope. Moreover, Section 2.4 presents an examination of dementia disease. Lastly, in Section 2.5, we highlight interesting datasets pertaining to dementia and AAL.

2.1. Neural Networks

One of the most remarkable assets of our society is the human brain, serving as the central core responsible for coordinating our actions, enabling us to experience sensations, and, most importantly, granting us the ability to think. Comprised of numerous specialized units known as neurons, the human brain facilitates the transmission of information through electrical signals. These neurons are instrumental in processing the vast amount of information we receive and form the foundational concept behind the creation of Artificial Neural Networks.

Artificial Neural Networks (Krogh, 2008) are a type of machine learning algorithm inspired by the structure and function of the human brain. With remarkable achievements across various domains, Artificial Neural Networks have become indispensable tools for a wide range of applications, including image recognition (Deorukhkar & Ket, 2022), natural language processing (Khurana et al., 2023), finance (Burrell & Folarin, 1997), and healthcare (Sordo, 2002). In Figure 2.1 is shown a basic Artificial Neural Network structure. During this work, we will use Artificial Neural Networks as the reference machine learning algorithm. In following sections, we will describe the most relevant structures for this project.

2.1.1. Convolutional Neural Networks

The CNN architecture is a specialized type of neural network designed specifically for Computer Vision tasks, including image and video recognition. CNNs consist of several key components, with the core being the convolutional layers. These layers are designed to apply a filter iteratively across the entire image, performing a dot product operation at each step. The resulting output, obtained from the various dot products, is referred to as a feature map or convolved feature. Subsequently, these feature maps are combined to extract more intricate features, enabling the network to recognize distinct patterns and elements within the input. A simplified illustration of a CNN structure is presented in Figure 2.2.

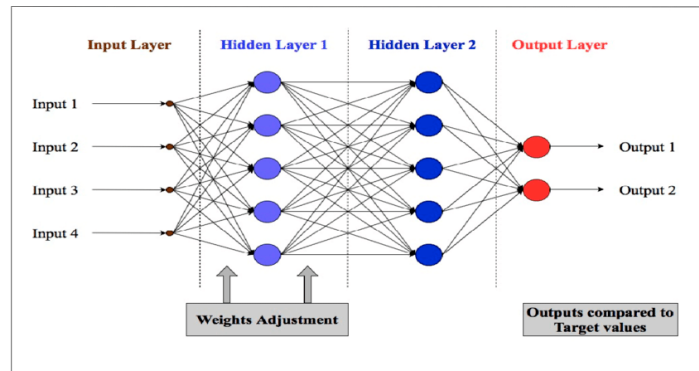


Figure 2.1: Basic Artificial Neural Network structure. Retrieved from Kumar et al. (2020).

While a wide range of CNN architectures exists, in practice, only a few variations have demonstrated superior performance. The following are the most noteworthy types of CNNs:

- **ResNet:** Residual Networks (ResNet) (He et al., 2015) introduced the concept of residual connections to address the degradation of network performance in deep neural networks with numerous layers. By mitigating the issue of vanishing gradients, ResNet enabled the successful training of networks with hundreds of layers.
- **AlexNet:** The AlexNet structure (Krizhevsky et al., 2017) was among the pioneering networks to adopt a deep architecture, leading to improved accuracy in visual classification tasks. It was the first network to utilize rectified linear unit (ReLU) activations, which accelerated the training process. Additionally, AlexNet employed dropout regularization and data augmentation techniques to mitigate overfitting.
- **VGGNet:** VGGNet (He et al., 2015) is a CNN architecture that prioritizes network accuracy as it deepens. It builds upon the foundations of AlexNet while employing a more streamlined and efficient structure with smaller convolution filters.
- **Inception:** Inception Net (Szegedy et al., 2014) was developed to extract features at different scales through the use of multi-scale convolution networks. It achieves this through an architecture composed of repeating components called Inception modules.
- **Xception:** The Xception Net (Chollet, 2016) enhances deep CNNs by employing Depthwise Separable Convolutions instead of Inception modules. Depthwise separable convolutions consist of two steps: depthwise convolutions and pointwise convolutions. This approach proves to be significantly more efficient than classical convolutions typically used in CNNs.

2.1.2. Recurrent Neural Networks

With the advent of CNN for image and video recognition tasks, there arose a need for variations of Neural Networks specifically designed to process sequential data. To address this, RNN were developed. RNNs are specifically designed for handling sequential data such as time series, speech signals, or text. They possess internal memory and feedback connections,

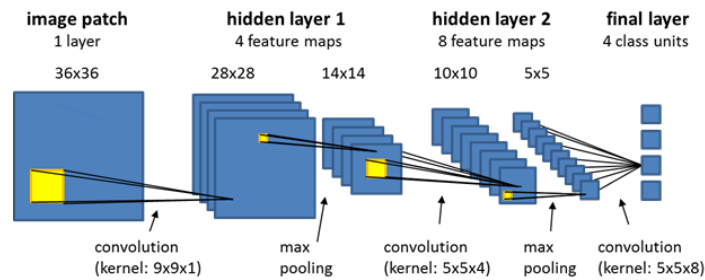


Figure 2.2: Basic intuition of CNN structure. Retrieved from Timilsina et al. (2019).

enabling them to store and utilize information from previous steps. The defining characteristic of RNNs is the presence of repeating modules, with each module processing one step of the input sequence. The output of each module serves as input to the subsequent module, while the hidden state of the module is updated to capture information from previous steps. This allows RNNs to accumulate information throughout the entire sequence remembering information of previous steps in subsequent ones. A basic illustration of the structure of an RNN is depicted in Figure 2.3.

While RNNs have made significant strides in tasks such as language modeling, sentiment analysis, machine translation, and speech recognition, they do have limitations when it comes to retaining information in long sequences. The vanishing gradients problem arises, wherein accessing information from earlier parts of a sequence becomes challenging as the sequence progresses. To overcome this limitation, various variants of RNNs have been developed. Next, we will introduce the most notable RNN architectures relevant to our work:

- **Long Short-Term Memory (LSTM):** The LSTM structure (Hochreiter & Schmidhuber, 1997) was specifically designed to address the vanishing gradients problem in RNNs. LSTMs include a memory cell that enables the retention of information over extended periods compared to traditional RNN structures. However, the computational complexity of LSTMs can result in longer training times, particularly when processing large datasets.
- **Gate Recurrent Unit (GRU):** The GRU structure (Cho et al., 2014) is a simplified version of the LSTM network. GRUs aim to capture long-term dependencies in sequential data while reducing the computational complexity associated with LSTM networks. While GRUs sacrifice some accuracy compared to LSTMs, they offer significantly reduced training times.

2.1.3. Transformers

The Transformers (Vaswani et al., 2017) architecture stands out as the leading model in contemporary times, consistently delivering exceptional results across various domains. It was specifically designed for sequence-to-sequence tasks, where the goal is to generate a text output based on an input text, such as in machine translation. Transformers are characterized by their utilization of self-attention mechanisms, which enable the network to selectively focus on different parts of the input sequence at each step and assign varying degrees of importance

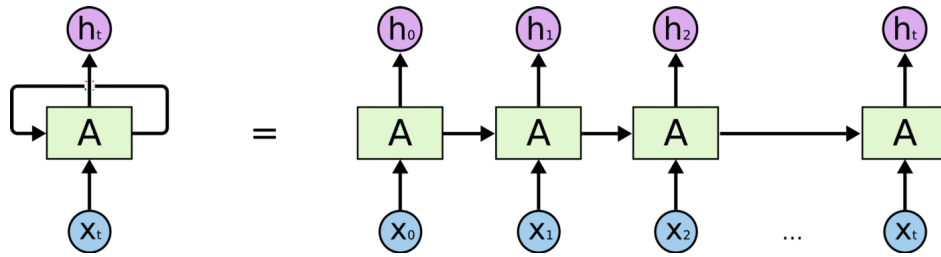


Figure 2.3: Basic intuition of RNN structure. Retrieved from Villamizar Torres & Lizarazo (2019).

to each part when making predictions. Further details about *attention* models are explained in Section 2.3.4.

Additionally, the Transformers architecture processes the entire sequence in parallel, allowing it to effectively utilize the contextual information from the entire input while it significantly reduces training times compared to other neural network structures such as LSTMs and GRUs. In Figure 2.4 is shown a basic structure for a Transformers Neural Network.

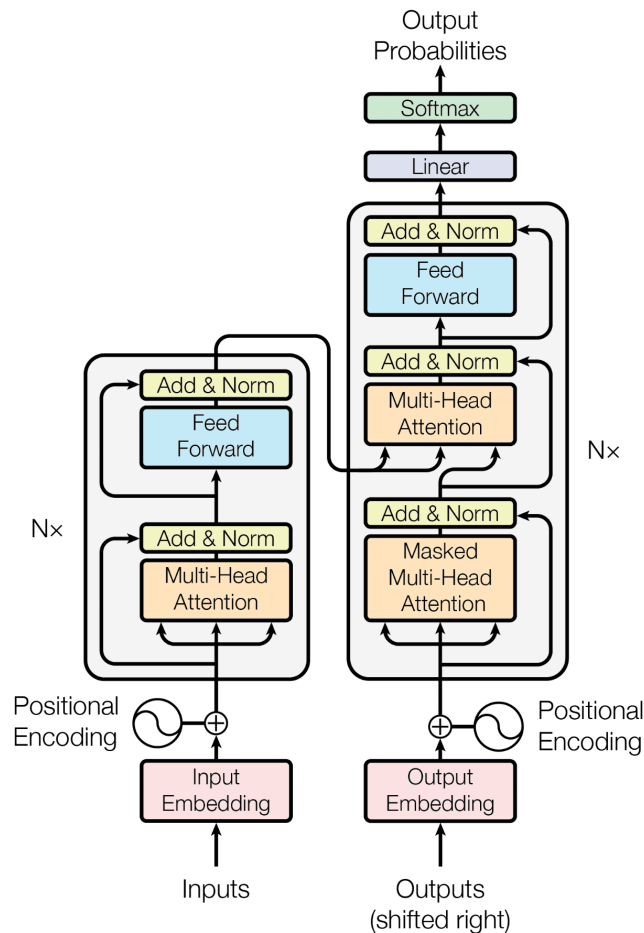


Figure 2.4: Basic intuition of Transformers structure. Retrieved from Mishev et al. (2020).

2.2. Captioning

Words and language serve as the fundamental tools for human communication, allowing us to share ideas with family, friends, partners, and groups of people. It is no surprising that the convergence of visual recognition and word processing techniques has led to the development of automatic description systems that are immensely valuable in today's society. The task of Captioning has gained significant importance as it contributes to remarkable progress in various domains, such as assisting visually impaired individuals (Manay et al., 2022) and automating the detection of alert situations in surveillance systems (Arriaga et al., 2017).

In recent years, we have witnessed a remarkable evolution in NLP in conjunction with Computer Vision and Object Recognition techniques. These advancements have resulted in a growing demand for generating descriptions from visual content inputs. These descriptions are commonly referred to as captions. Captions are concise representations of the content or events depicted in the provided visual input, thus giving rise to the task of Captioning. Given the diverse nature of visual content, there exist different types of Captioning tasks. In the following subsections, we will introduce Image Captioning and Video Captioning, with the latter being the primary focus of this chapter.

2.2.1. Image Captioning

The task of Image Captioning emerged as one of the earliest approaches based on the concept of generating captions for images. It involves obtaining concise natural language descriptions from input images, as illustrated in Figure 2.5. Image Captioning has been extensively studied within the DL community, highlighting its wide range of applications. Recent interesting research in the field of Image Captioning has explored the use of models to enhance the clarity of highly blurred images (F. Chen et al., 2019). Despite being a well-studied task, Image Captioning still presents some challenges (Deorukhkar & Ket, 2022), which will continue to be addressed over time. The challenges in Image Captioning can be summarized as follows:

- **Boundaries estimation:** Identifying visual details of objects becomes challenging when object boundaries become indistinct or ambiguous.
- **Learning intermediate representations:** Establishing effective intermediate representations between the visual and natural language domains is a critical issue in image-to-text conversion techniques.
- **Ranking of visual elements:** Determining the importance of objects and their specific features in an image is crucial for generating accurate and detailed textual descriptions.

While Image Captioning is not the primary focus of our work, it is important to expose these challenges. In the next section, we will delve into Video Captioning, which shares a similar fundamental architecture to Image Captioning.

2.2.2. Video Captioning

The outstanding success of Image Captioning techniques has led to the adoption of similar approaches in the field of videos, leading to the central concept explored in this chapter:



Figure 2.5: Examples of image captioning descriptions.

Video Captioning. Video Captioning is defined as a DL task that focuses on generating natural language descriptions of the visual content within a given input video. In essence, Video Captioning encompasses the techniques employed to extract textual information from the features obtained through Computer Vision mechanisms for video analysis. This process involves two main tasks: first, comprehending and recognizing the visual content of the video, and second, grammatically describing this content.

While humans can easily understand the contents of a video, the same cannot be said for computers. Video Captioning presents significantly higher computational complexity compared to Image Captioning due to additional factors that must be considered. The following key aspects highlight the challenges and complexities specific to Video Captioning:

- **Selective importance of objects and actions:** Not all objects or actions in a video are relevant for generating accurate descriptions. Unlike images, videos contain a wide variety of objects and actions, but only the relevant ones must be considered.
- **Motion and relational understanding:** Video Captioning methods must capture the motion, relationships, causality, and trajectories of objects within the video. In contrast, images represent static scenes without object motion.
- **Variable event lengths and overlaps:** Events in videos can have different durations and may overlap with each other. In images, events are fixed and do not exhibit temporal variations.
- **Consideration of temporal features:** Video Captioning requires attention to both spatial and temporal features. In images, only spatial-based features need to be extracted.

Video Captioning has undergone various advancements, resulting in different types of tasks based on the level of detail present in the generated descriptions. The distinctions between


	
Dense video captioning	A boy is riding a bicycle. He loses his balance and falls on the ground. He gets back up and starts riding again.
Single Sentence captioning	A boy in red t-shirt is riding a bicycle.

Figure 2.6: Difference between Single sentence Video Captioning and Dense Video Captioning. Retrieved from Islam et al. (2021).

these tasks are illustrated in Figure 2.6. The following types of Video Captioning tasks can be identified:

- **Single sentence Video Captioning:** This is the original Video Captioning task, where the objective is to summarize the entire video using a single sentence. However, this approach often leads to descriptions that lack detailed information and fine-grained understanding of the video content.
- **Dense Video Captioning:** This task emerged to address the need for more detailed descriptions in Video Captioning (Krishna et al., 2017). In Dense Video Captioning, multiple sentences are used to provide a richer and more informative description of the video. It allows for capturing fine-grained details of multiple overlapping events with varying durations. Dense Video Captioning holds great promise for the future of video description generation. Several improvements have been made in this area, such as the work by Deng et al. (2021), which reverses the typical "detect-then-describe" scheme and proposes a top-down approach. It first generates paragraphs from a global view and then grounds each event description to a video segment for detailed refinement.

In the upcoming sections, we will demonstrate how a pure Video Captioning model can be adapted to produce longer descriptions by training it on data with denser captions. Our experiments will show that the length of the generated descriptions depends on the pretrained checkpoint of the model used. However, it is important to note that pure Video Captioning models cannot achieve the same level of detail as Dense Video Captioning models. Further research is required to explore whether Dense Video Captioning models based on Transformers can achieve the same level of precision as pure Video Captioning models.

2.3. Video Captioning in depth

The Video Captioning task encompasses various techniques that utilize Computer Vision approaches to generate natural language descriptions from input videos. In this section, we will explore this task in detail. Firstly, we will describe the fundamental structure of a Video Captioning model in subsection 2.3.1, this will provide an overview of the key components

and their interactions within the model. Then in subsections 2.3.3, 2.3.2 and 2.3.4 each part of the main structure will be further discussed.

2.3.1. Fundamental Structure

Since the early days of Video Captioning research (Koller et al., 1991), various approaches have been developed to efficiently and accurately tackle this task. As we discussed in previous sections, Video Captioning involves two main tasks: recognizing objects or events in videos and expressing this information in grammatically correct sentences. Therefore, the fundamentals of Video Captioning are typically based on two-stage solutions. In the past, a prominent architecture for Video Captioning involved using the concept of Subject, Verb and Object (SVO) relationships as the first stage for visual content detection, followed by a template-based model for text generation (Kojima et al., 2002). However, with the advent of DL and Artificial Neural Networks, the landscape of Video Captioning changed significantly. Neural Networks revolutionized the field of Machine Learning, including Video Captioning, so the traditional SVO and template-based methods became obsolete as Neural Networks were applied to this task.

The most commonly used two-stage Neural Network approach is based on an *Encoder-Decoder* framework. In this framework, the visual content is first input to a 2-D/3-D Convolutional Neural Network (CNN), which serves as the "encoder" and extracts features from the visual content. The output of the CNN is then fed into a RNN, which acts as the "decoder" and generates the final description sentence in natural language based on the visual features. Figure 2.7 provides an overview of the most widely used Video Captioning architectures.

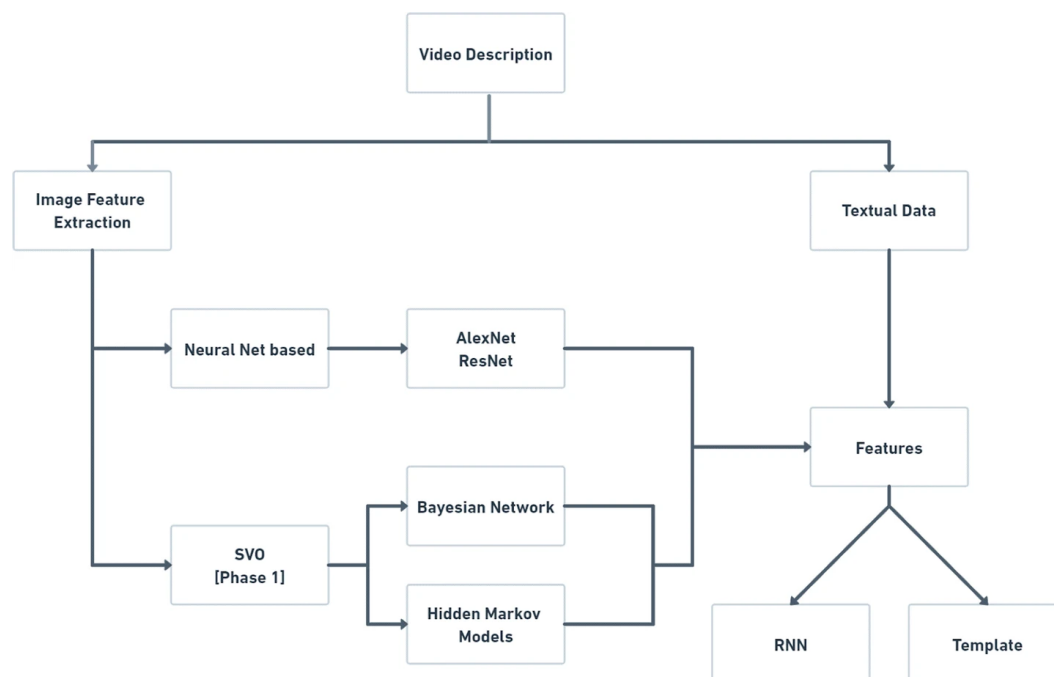


Figure 2.7: Resume of outlined Video Captioning structures. Retrieved from Jain et al. (2022).

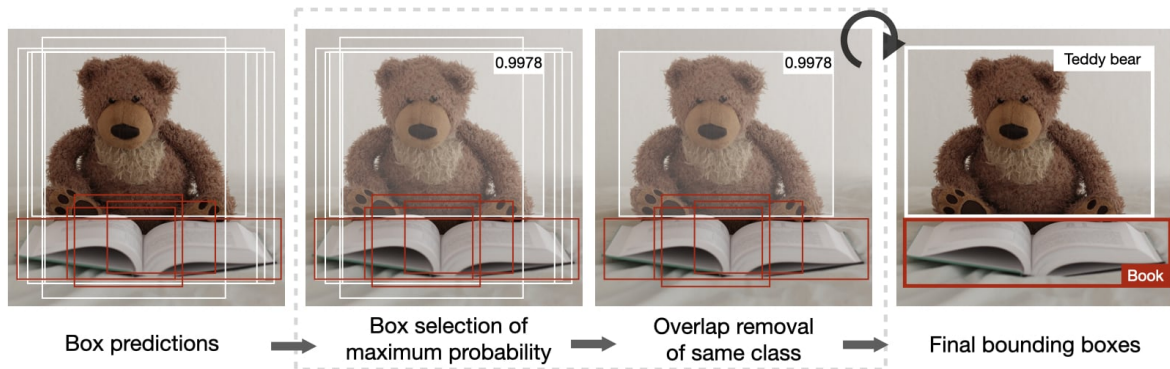


Figure 2.8: Example of overlapping bounding boxes removals. Retrieved from <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks>.

2.3.2. Encoding stage (CNN)

The first stage of the description generation process in Video Captioning focuses on the Computer Vision aspect of the task. In this stage, the video content is input to a CNN for visual analysis of the video frames and extraction of features related to recognized events, objects, and their relationships. The CNN is responsible for detecting various items and actions present in the entire video. To achieve this, the video is divided into individual frames, which serve as inputs to the network. Once the frames are fed into the neural network, the detection of video contents occurs through the different convolutional layers, employing various methods. One commonly used method for content detection is Bounding Box detection, where bounding boxes are created around regions where objects are detected. Due to the overlapping of different objects in the frames, multiple bounding boxes are often detected. To address this issue, the first detection of objects is often treated as *Proposal boxes*. These proposal boxes then are provided to a filtering mechanism to retain the most representative boxes, which helps eliminating redundant or less relevant bounding boxes. This procedure can be seen in one notable Video Captioning work that achieved remarkable results (Jin et al., 2019). In this, the authors introduced the *OCA* algorithm, which utilizes proposal boxes to classify relevant boxes. The concept of proposals can also be applied to temporal features, allowing for more precise detection and filtering of events (Krishna et al., 2017). Figure 2.8 provides an example of bounding box filtering, demonstrating the process of selecting relevant boxes for further processing in Video Captioning.

The encoding stage in Video Captioning typically employs a CNN to extract visual features from the input video. While this is the most common and basic scheme, more complex approaches have also been explored, resulting in significant performance improvements. In some works, the encoder is composed of a RNN (Jin et al., 2019), which enhances the main CNN by providing temporal context features. Various schemes with different components have been developed and tested, but the most famous and widely used components today are the *attention* units. These units can be found not only in the encoder phase (Zhou et al., 2018) but also in the decoder phase (Pei et al., 2019). The concept of *attention* models will be discussed in more detail in the following sections.

The key components of the encoding stage are the CNN units. It is common to encounter

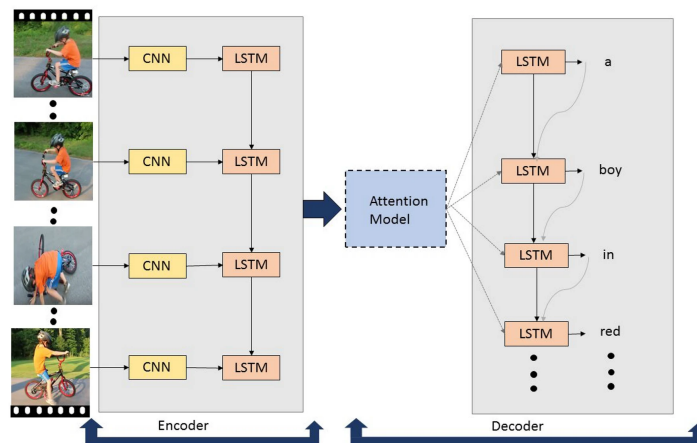


Figure 2.9: Example of possible Video Captioning framework. Retrieved from Jain et al. (2022).

more advanced variations of CNN units such as ResNet, Inception Net, Xception Net, or other similar structures like an Regions with Convolutional Neural Network features (R-CNN) based unit. These advanced CNN architectures enable the extraction of more sophisticated and high-level visual features from the input video, contributing to improved performance in Video Captioning tasks.

2.3.3. Decoding stage (RNN)

The decoding stage is the second phase of the description generation process. In this stage, the visual features extracted from the encoder are passed as input to an RNN decoder, which aims to generate an accurate natural language description of the provided video. Prior to the decoding stage, the visual features are typically encoded into a fixed-length vector. This encoding step reduces the dimensionality of the features while capturing the most relevant information. The encoded vectors serve as input to the sequence model, which decodes the visual features into a natural language description. The decoder is commonly composed of RNN units such as GRU and LSTM, or it can be based on Transformers.

Similar to the encoder stage, the decoding stage often incorporates the *attention* mechanism in state-of-the-art works. For instance, in Li et al. (2019), an attention-based mechanism is used to capture temporal information during decoding.

A possible example of a complete Video Captioning encoder-decoder framework can be observed in Figure 2.9, where an interaction with an attention unit is also depicted. This Figure shows the complete flow of information that composes a Video Captioning pipeline.

2.3.4. Attention models

The attention mechanism is based on the principle that focusing on the most important parts leads to achieving the highest possible results, regardless of the task at hand. In Video Captioning frameworks, attention models have emerged as the most innovative neural network structures in recent years. These models aim to highlight the most relevant parts of the input by assigning weights to each element that constitutes the input. Each weight

represents the importance of a specific element for a particular prediction. Subsequently, the model utilizes these weights to compute a weighted sum of the inputs, which serves as the final representation that is further processed by the network. This dynamic allocation of attention allows the model to adaptively concentrate on the most significant aspects of the input, instead of relying on a fixed representation that may not be suitable for all instances.

The concept of attention was initially introduced in Bahdanau et al. (2014), and it served as the fundamental concept for the development of Transformers, which have become the most influential neural network structures in present times. Figure 2.10 illustrates an example of the attention model applied to a video frame, demonstrating its ability to emphasize the relevant regions of the input.

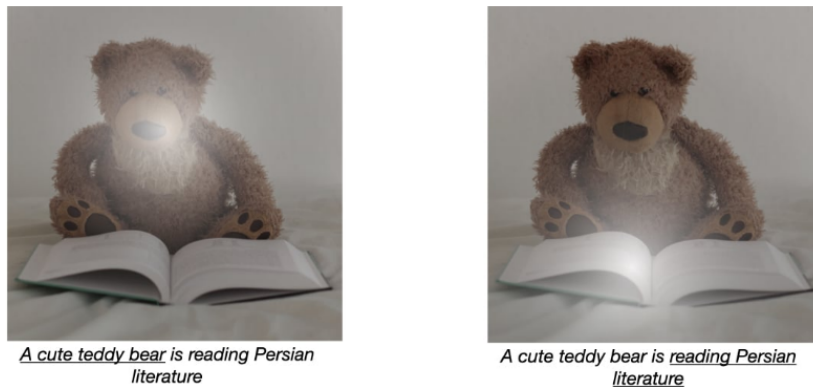


Figure 2.10: Attention applied to generate a video frame caption. Retrieved from <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks>.

2.3.5. Indoor scene captioning

The indoor scene recognition is a highly studied field inside Machine Learning (ML) scope, specially inside the contexts of robotics and works aimed to create helpful systems to visually impaired people. Classical scene classification was based on hand-crafted features and so centered in the object detection to determine the kind of scenes. Because of the limitation of studies for using data labelled on 2 dimensions, other authors made progresses introducing the usage of 3D data in the scene recognition (Huang et al., 2020) which leads to better results that using 2D data, specially in robotics scope. The introduction of neural networks in the ML paradigm provoked that this task evolved to use these structures. We can see this evolution in some recent works (Afif et al., 2020) on which authors rely on the neural network scaling using rethought CNN.

Due to the necessity of extracting more useful information about scenes, authors began to not only be aimed in classifying the scenes but also in obtaining a further description which represents in natural language what can be seen in the scene. Some authors (Fudholi & Nayoan, 2022) base its approaches in the customization of known datasets as MSCOCO (T.-Y. Lin et al., 2015) introducing new ground-truth features for the provided captions. Others (D. Lin et al., 2015) centered their efforts in creating richer indoor multi-sentence captions by the usage of a 3D visual parsing system and a text generation algorithm that takes into

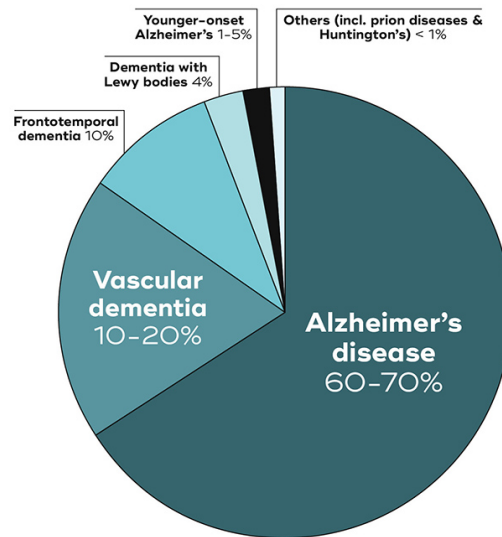


Figure 2.11: Average percentage of the different types of dementia.

account coherence between sentences. This task has also gained importance in the realm of ambient assisted living, where real-time captioning services are being developed to enhance the safety of the elderly.

2.4. Dementia

Dementia is a widespread disease that although it is commonly discussed, is crucial to recognize its importance and increasing prevalence. Dementia refers to a syndrome characterized by the progressive decline of cognitive functions such as memory, visuospatial abilities, executive functions, and thinking (World Health Organization, 2021). The impact of dementia is substantial, with an estimated 78 million people projected to have dementia by 2030 and 139 million by 2050 (Alzheimer's disease international, 2011). Additionally, dementia ranks as the seventh leading cause of death and is a significant contributor to disability and dependency among older individuals worldwide (World Health Organization, 2021). While there are various types of dementia, Alzheimer's disease is the most prevalent form, accounting for approximately 60-70% of all dementia cases globally. Figure 2.11 provides a summary graph illustrating the distribution of dementia types.

Early detection of dementia is crucial to mitigate its severe impacts on affected individuals. However, identifying dementia at an early stage is challenging, as noticeable symptoms typically manifest only when neuronal damage has already spread extensively and become irreversible. Consequently, current research efforts are dedicated to developing methods capable of detecting dementia in its early stages. When a patient is identified during this phase, it is often referred to as Mild Cognitive Impairment (MCI), which is considered a transitional stage between normal aging and early dementia. It is important to note that not all individuals with MCI progress to dementia, but they are at an increased risk compared to those without MCI. Early diagnosis plays a pivotal role in slowing down the progression

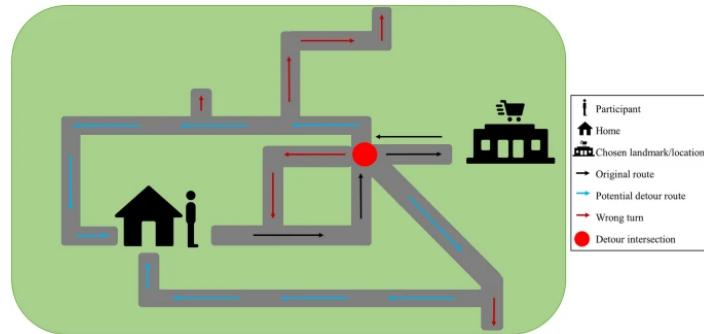


Figure 2.12: Map of VR test for dementia patients. Retrieved from Puthusserypady et al. (2022).

to dementia.

Various approaches are being explored for the detection of dementia. Neuroimaging, particularly through techniques like Magnetic Resonance Imaging (MRI), is commonly employed because represents the most reliable approach but is not always recommended due to its invasiveness, behavioral analysis often utilizing sensors to monitor gaits, emotion analysis using eye tracking or facial gesture monitoring, and cognition tests such as the Mini Mental State Examination (MMSE) are among the prevalent methods. However, these techniques are continuously evolving to develop more effective and less invasive detection methods. Technologies like Virtual Reality (VR) or Virtual Environments (VE) are being harnessed to create immersive contexts where patients can perform cognitive and behavioral tasks, providing researchers with valuable insights into their mental and cognitive states (Fernández Montenegro et al., 2020). In Figure 2.12 is shown a map from a VR test on which patients must navigate to a chosen landmark or location in their neighbourhood that they commonly visit using their usual route.

2.5. Dementia and AAL datasets

With the interest growth in dementia and AAL research fields, corporations and research laboratories have started creating datasets with the objective of developing intelligent systems that can analyze this information to make predictions about the mental health of patients or provide assistance to dependent individuals in their daily lives. Examples of such corporations include the University of Michigan Health and Retirement Study (HRS) and the National Health and Aging Trends Study (NHATS). These studies conduct annual in-person and in-depth interviews with individuals aged 65 and above, generating a rich and multidisciplinary dataset that researchers can utilize to address important questions about the challenges and opportunities associated with aging.

Despite the significance of these research fields, there is a scarcity of available datasets that are specifically tailored to the objectives we seek to achieve. However, we will outline some of the most interesting datasets comprising various types of data, which can be employed in the development of systems focused on advancing dementia recognition and AAL research. While our study does not primarily focus on dementia, we will mention relevant datasets related to this topic, as it represents one of the primary motivations for the development of this thesis.

2.5.1. DementiaBank

DementiaBank¹ is a database of multimedia interactions for the study of communication in dementia. It is composed by different datasets which contains speech and language data, specifically transcriptions of speech from people with dementia as well as healthy control participants, in some different languages: English, Spanish, Mandarin and Taiwanese. The transcriptions contained in the datasets include information about the speaker's language ability, including measures of word production, sentence comprehension, and grammatical abilities. These datasets also includes demographic information about the participants, such as their age, education level, and gender.

Inside the datasets included in DementiaBank can be seen the results for a variety of different tests. Next will be mentioned the most relevant tests that comprises the different DementiaBank datasets:

- **The Boston Naming Test:** Test where the word recall ability of the patient is measured. Data of this test can be found on the *WLS* dataset (Herd et al., 2014), which contains audio data and its associated transcriptions.
- **The Picture Description Task:** Test where the patient does a description of a picture, providing a measure of their language ability and cognitive function. Data for this test can be found on the *Kempler* dataset (Kempler et al., 1987), which contains only audio data. In Figure 2.13 is shown one of the most usual pictures presented to dementia patients to be described.
- **The Sentence Comprehension Task:** This test involves giving responses to questions about sentences, providing a measure of their language comprehension abilities. Data for this test can be found on the *Pitt* corpus (Becker et al., 1994), which contains audio data and its associated transcriptions.

2.5.2. Dem@Care

Dem@Care (Karakostas et al., 2016) is a bench of dementia-aimed datasets which contain different formats of data, from video and audio recordings to physiological data provided by unobtrusive sensors. Moreover, they also include other kinds of data like information from sleep, motion and plug sensors. The gathering of the data took place in the Greek Alzheimer's Association for Dementia and Related Disorders in Thessaloniki, Greece and in participants' homes.

One of the main activities of Dem@Care focuses on the analysis of daily activities of the people with dementia in their domestic environment via the extraction and processing of features describing their actions and the context in which they occur. These features were extracted from various raw contextual and wearable sensors and their interpretation in terms of activities and events. Moreover, this project provides a set of complementary features that consists on visual perception for the detection and recognition of situations of interest, audio-based affective analysis for characterizing the person's behavioural, mental and emotional state, and instrumental activities monitoring for characterizing the person's

¹<https://dementia.talkbank.org/>

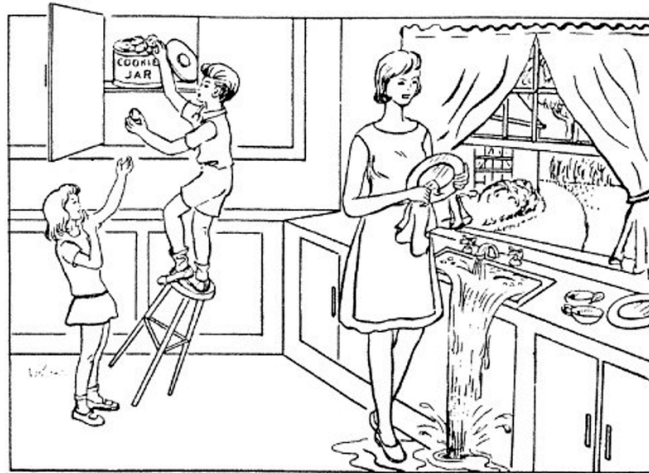


Figure 2.13: This picture is called the Cookie Theft, it is a common test image used in picture description tasks.

actions and the organization of detected events into a life-log. The objective of the project is to provide an intelligent knowledge structure suitable for making decisions depending on the analysed behaviour of the person in particular. In this way support systems for persons suffering dementia can be constructed, helping them in completing tasks of their daily life. In Figure 2.14 one of the test environments used during the recording of the dataset is shown.

2.5.3. Charades

Charades (Sigurdsson et al., 2016) is a large-scale dataset comprising of 9,848 videos of daily home activities, with each video having an average duration of 30 seconds. The dataset was created using crowdsourcing, with the entire process of video creation, script writing, recording, and annotation being developed in a distributed manner. Charades was created by 267 people from three different countries through Amazon Mechanical Turk service.

The authors of Charade aimed to create a dataset of videos that are as casual and realistic as possible. For this reason, they used crowdsourcing for the entire construction process to maintain the bias of each person involved in the creation process towards the activities, thus preserving the essence and nature of the daily tasks of each person.

A total of three stages were involved to create the dataset:

- **Scripts generation:** First some workers created the scripts for the recording of the subsequent videos. Each script has to follow some guidelines established by the authors. This guidelines are available on the official dataset publication.
- **Video generation:** Then the previous scripts are given to workers and they were requested to record a video interpreting the given scripts with a duration of 30 seconds.
- **Annotation:** Finally different workers were given some videos and they were requested to describe what they saw on the videos. After that, other workers were asked to create a list of object and actions that are present on each of the videos. With the actions also were attached a label with the starting and ending point of each of them.



Figure 2.14: Example of test environment for participants of Dem@Care project.

All of this process resulted in a dataset comprising 27,847 video descriptions, 66,500 temporally localized intervals for 157 action classes, 41,104 labels for 46 object classes, and 15 types of indoor scenes. In figure 2.15 can be seen a sample frame of the dataset.



Figure 2.15: Charades dataset sample frame.

2.5.4. ETRI-Activity3D

Research on elder care robots has emerged as a highly investigated topic worldwide. Despite the numerous corporations and laboratories exploring this field, there is currently no publicly available dataset that provides suitable data for robots to understand and recognize the daily activities of human users. To address this gap, the ETRI-Activity3D dataset (Jang et al., 2020) was created as the first RGB-D large-scale dataset specifically focused on the daily activities of the elderly for human care robots.

The dataset was collected using Kinect V2 sensors positioned at heights of 70 cm and 120 cm, which align with the typical height of human-care robots. It consists of three synchronized

data modalities: RGB videos, depth maps, and skeleton sequences. The RGB videos have a resolution of 1920 x 1080, while the depth maps are stored at a resolution of 512 x 424 frame by frame. To gather the data, 100 different actors were recruited, comprising two distinct groups: an elderly group of 50 individuals ranging from 64 to 88 years old (with an average age of 77), and a group of young individuals in their 20s (with an average age of 23).

Within this dataset, researchers can find a collection of 55 different daily life actions. These actions were selected based on an investigation of the daily behaviors from morning to night of 53 elderly individuals with an average age of 70. The set of 55 actions was defined based on the most frequently observed activities. Overall, the dataset encompasses a total of 112,620 samples, including RGB videos, depth maps, and skeleton sequences. In Figure 2.16 some RGB and associated skeleton sequences video samples of the dataset are shown.

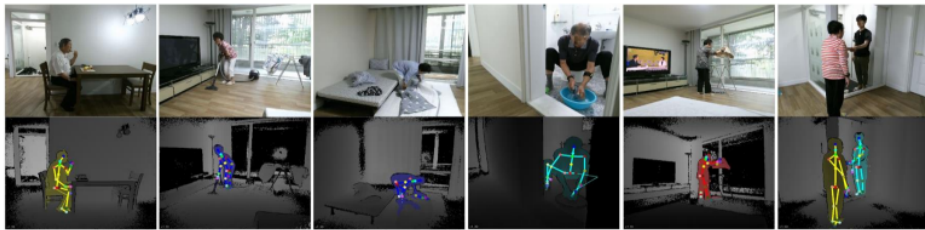


Figure 2.16: ETRI dataset sample frames.

2.5.5. VirtualHome

As stated before, datasets with features and annotations suitable for our purposes are scarce. When dealing with this problem researchers tend to resort for using as part of the trainset synthetic data. This is where VirtualHome (Puig et al., 2018) comes into play. VirtualHome is a simulator which allow us to create a large-scale video dataset in a household 3D environment.

In VirtualHome, the simulator is controlled by programs, which consist of instructions specifying the desired actions, objects to interact with, and actors performing the actions. The output of these programs is a simulation where the specified action is carried out by the actor. Additionally, the simulator provides dense ground-truth information such as semantic segmentation and depth, among others. Dataset creation from VirtualHome involves recording the output simulations. The authors of the simulator have also developed models that enable the creation of these programs from natural language and video demonstrations within the simulator. The simulator utilizes the Unity 3D game engine as its backbone for generating the simulations. It includes six furnished homes and four rigged humanoid models from the Unity Assets Store web. Each home contains an average of 357 object instances, classified into 30 different object types. To ensure visual diversity, the authors collected at least three different models per object class.

Using the VirtualHome simulator, the authors proposed two datasets. The ActivityPrograms dataset covers 75 atomic actions and 308 objects derived from 2,821 different programs. The VirtualHome Activity Dataset, on the other hand, consists of 5,193 different programs focused solely on the 12 most frequent household activities. The simulations for both datasets were generated by randomizing factors such as home selection, agent selection, camera placement, object placement, initial agent location, action speed, and choice of ob-

jects for interactions. Examples of frames and descriptions from both datasets can be seen in Figure 2.17.

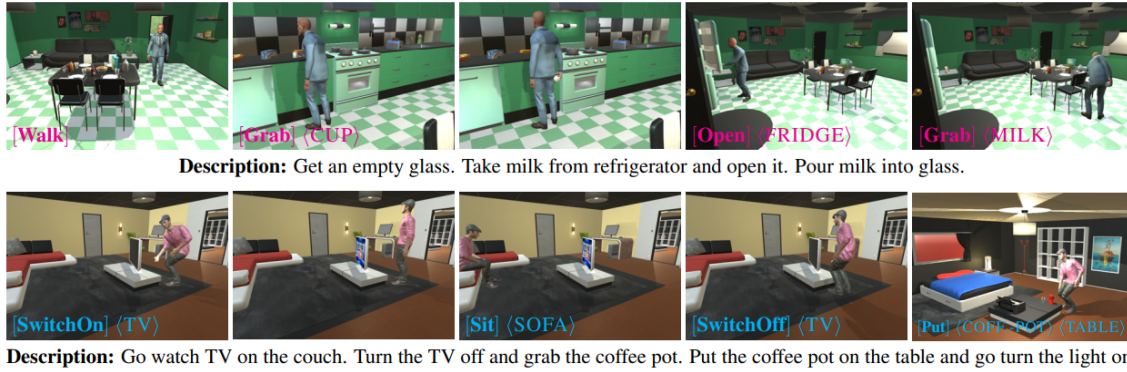


Figure 2.17: Simulation frames and descriptions from the ActivityPrograms and VirtualHome Activity datasets.

3. Methodology

In this third chapter, we will review all the resources and methods used throughout the entire experimentation process. These resources encompass hardware, software, and data materials collected for our purpose. The section is structured as follows: Section 3.1 provides a summary of the hardware utilized. Section 3.2 presents a synopsis of the software employed. In Section 3.3, we discuss various machine learning libraries of interest. Finally in Section 3.4 we will specifically highlight the machine learning libraries ultimately chosen for implementation.

3.1. Hardware

Solid hardware support is crucial for the development and testing of a modern deep learning system architecture. In order to expedite the experimentation process, our main workstation support was provided by the 3DPerceptionLab at the University of Alicante, referred to as Asimov.

In Table 3.1 is shown all specifications regarding the hardware support that provides Asimov. It must be outstated that although it has its own hard disk support, it proved insufficient to store all the required datasets. Consequently, Asimov is connected to a Network Attached Storage (NAS) with a total capacity of 25.3TB. Remote access to Asimov is established through Secure Shell (SSH) using an ISL network. ISL is a networking protocol utilized to create network tunnels from personal computers to the university network, granting us access. Authentication on the server is provided by the RSA public key protocol.

3.2. Software

In addition to reliable hardware support, a robust software platform is essential to facilitate the required experimentations while minimizing any potential incompatibilities and conflicts among the technologies used in various tests. To address this, we will introduce the utilization of Docker in this section. Docker enables us to create distinct test and development environments tailored to our evolving needs. This approach ensures flexibility and helps mitigate compatibility issues that may arise.

Docker¹ is an open-source platform designed to simplify the deployment of application environments for developers. It promotes the use of containers to build, package, and deploy applications, effectively addressing environment-related issues during runtime. Docker significantly reduces software dependency problems by providing isolated containers that maintain their own software and dependency versions, independent of the native operating system. A Docker container is created from an image. An image is a lightweight and portable package that contains all the necessary software and dependencies to run an application. Images are

¹<https://www.docker.com/>

Asimov	
Motherboard	Asus X99-A Intel X99 Chipset 4x PCIe 3.0/2.0 x 16(x16, x16/ x 16, x16/ x 16/ x 8)
CPU	Intel(R) Core(TM) i7-5820K CPU @3030GHz 3.3 GHz (3.6 GHz Turbo Boost) 6 cores (12 threads) 140 W TDP
GPU (0)	NVIDIA Titan X 3584 CUDA cores 12 GB of GDDR5 Video Memory PCIe 3.0 250 W TDP
GPU (1)	NVIDIA GeForce GTX Titan X 3072 CUDA cores 12 GB of GDDR5 Video Memory PCIe 3.0 250 W TDP
RAM	4 x 8 GB Kingston Hyper X DDR4 2666 MHz CL13
Storage (Data)	(RAID 0) 2x Seagate Barracuda 7200rpm 3TB SATA III HDD
Storage (OS)	Samsung 850 EVO 500GB SATA III SSD

Table 3.1: Specification details of Asimov.

generated from Dockerfiles, which consist of a set of instructions specifying the characteristics and features required to create the image. Figure 3.1 illustrates an example of creating a new container from one of the Docker images employed in our experimentation. The command demonstrates our intent to utilize one GPU and create the container based on the image named *javiro01/swinbert-evaluation*. Furthermore, the *-rm* flag signifies that the container should be removed once the session is terminated. The *-it* flag indicates that an interactive bash terminal should be opened within the container.

```

1 docker run --gpus '"device=0"' --rm -it \
2   --volume="$(pwd):/workspace:rw" \
3   --workdir="/workspace" \
4   javiro01/swinbert-evaluation bash

```

Figure 3.1: Sample command to run a container through the docker CLI.

3.3. Machine learning frameworks

Currently, there is a wide range of machine learning frameworks available, each serving specific needs and purposes. In this section, we will provide an overview of the most renowned frameworks within our scope. Subsequently, we will focus on explaining the frameworks that

were utilized in our work.

3.3.1. TensorFlow

TensorFlow (Abadi et al., 2016) is a widely used open-source machine learning library developed by Google. It supports the creation and training of deep learning models and is implemented in Python and C++. TensorFlow offers several key features, including integration with the Keras² API, distributed computing support for training and deploying models across multiple devices, efficient handling of large multidimensional tensors, and the ability to execute same code on both GPU and CPU. Furthermore, TensorFlow provides TensorFlow Serve, its own framework for deploying models in production environments. It also offers APIs for development in various programming languages, including Python, C++, Haskell, Java, Go, and Rust.

3.3.2. PyTorch

PyTorch (Paszke et al., 2019) is a machine learning framework based on the classical Torch library, which is widely used for scientific computing. Developed by Meta AI, PyTorch is extensively utilized in the fields of natural language processing and computer vision. It is implemented in Python, C++, and CUDA, providing support for both Python and C++ programming languages.

PyTorch offers several key advantages. Firstly, it leverages GPU acceleration and data parallelism for efficient tensor computations. It also facilitates the creation of deep neural networks through automatic differentiation. Additionally, PyTorch provides TorchScript³, a tool for saving and optimizing models, and enables dynamic graph computations, allowing developers to modify the behavior of neural networks on the go. One of the standout features of PyTorch is its pythonic approach, which makes it highly user-friendly. Moreover, the framework is complemented by TorchServe⁴, a production-ready framework for deploying models easily.

3.3.3. HuggingFace Transformers

HuggingFace (Wolf et al., 2020) is a collection of open-source resources designed for the development and maintenance of various types of machine learning systems, spanning from computer vision to text or audio processing. Within this collection, we will focus on HuggingFace Transformers. Transformers is a library that offers an API for developing new machine learning models or utilizing pretrained models available in their repository. One of the most notable and powerful features of this API is the inclusion of *pipelines*. These interfaces allow for streamlined inference for specific machine learning tasks, requiring only a few lines of code and providing a high-level and intuitive coding experience.

HuggingFace Transformers has become a valuable resource for the open-source machine learning community by providing easy-to-follow references for data processing, model training, and making inferences. This library is built on the foundations of PyTorch and Tensor-

²<https://keras.io/>

³<https://pytorch.org/docs/stable/jit.html>

⁴<https://pytorch.org/serve/>

Flow, and during development, you have the flexibility to choose which backbone implementation you prefer to use.

3.3.4. OpenCV

OpenCV (Culjak et al., 2012) is an open-source machine learning library focused on computer vision and image processing. Developed by Intel, it is primarily implemented in C and C++. This framework boasts several key features that have established OpenCV as a prominent computer vision library within the field of machine learning. OpenCV offers an extensive range of functions for various purposes, including face recognition, feature detection, and camera calibration, among others. Its comprehensive set of functions makes it a reference library for computer vision tasks. Additionally, OpenCV is highly efficient and fast, benefiting from being developed in the lightweight programming language of C++. Furthermore, OpenCV supports extensibility, allowing developers to integrate their own algorithms into the library. This flexibility has fostered a large community of users who contribute tutorials and documentation, further enhancing the available resources for learning and development. The ease of use and versatility of OpenCV are exemplified by its compatibility with multiple programming languages such as C++, Python, and Java. Overall, OpenCV's rich functionality, performance, extensibility, and wide language support have positioned it as a popular choice for computer vision and image processing tasks in the machine learning community.

3.3.5. SpaCy

SpaCy⁵ is an open-source library specifically designed to address common challenges encountered in NLP tasks across more than 60 languages. This versatile library offers a wide array of pretrained models tailored to various text processing tasks, including tokenization, part-of-speech tagging, dependency parsing, and named entity recognition. One noteworthy feature of SpaCy is its efficient memory management, making it ideal for processing large volumes of information without encountering performance issues. The framework has been meticulously engineered with Python and Cython, enabling effective scaling of memory management. Additionally, SpaCy provides a user-friendly API for seamless integration with Python-based workflows.

SpaCy stands out as an invaluable tool in the NLP domain, offering comprehensive solutions for multilingual text processing. With its extensive range of pretrained models and robust memory management capabilities, it has become a preferred library for tackling complex NLP tasks efficiently.

3.3.6. Gensim

Gensim (Řehůřek & Sojka, 2010) is an open-source Python library specifically designed for representing documents and plain text as semantic vectors using unsupervised machine learning algorithms. It offers a wide range of NLP algorithms that enable the retrieval of embeddings from text. Notable algorithms within Gensim include Word2Vec, FastText, Latent Semantic Indexing, and Latent Dirichlet Allocation, among others. The library also provides a training API for fine-tuning large-scale NLP models.

⁵<https://spacy.io/>

Gensim includes pretrained models that can be utilized for various NLP tasks, particularly those related to semantic understanding. These models can be pretrained in different specific domains such as legal or health. Furthermore, Gensim has been developed in C to leverage its execution speed and employs parallel routines to optimize processing times. Moreover, the library adopts a data-streamed approach in its algorithms, mitigating issues of running out of RAM when processing extensive corpora.

3.4. Selected technologies

In the above subsections a summary of a collection of the most used machine learning libraries and frameworks has been developed. Now we will mention the ones which were finally used and why.

When a researcher begin with a machine learning project implementation the first question that must be resolved is which is the main backbone framework that it is going to be used, the majority of them ends up in Pytorch or in TensorFlow. While TensorFlow offers better visualization and debug tools to train models, Pytorch stands out more in the area of data parallelism providing an API on which the distributed processing is automatically implemented. Pytorch has an API on which due to its pythonic approach its usage is so simple, by the other side TensorFlow is more focused in researchers with a more advance knowledge in deep learning techniques which wants more control over all the processes. For our purposes we have chosen Pytorch as our main backbone not only due to its simplicity and its shorter learning curve, but also because it is incorporating from time to time new functionalities which are reducing the difference between this and its main competitor TensorFlow. As we can see in figure 3.2 Pytorch is gaining popularity in this last years inside the research community. It must be outstanded that other frameworks as Keras or Theanos are also used in the deep learning community depending on the needs of the project, but in this study we only included the more relevant ones.

Once we have selected the primary backbone framework, we can choose additional libraries based on our specific needs. Since our focus is primarily on computer vision and NLP tasks, we will utilize the OpenCV library for computer vision problems and SpaCy for NLP-related tasks. During experimentation, we will observe that OpenCV is integrated into the backbone implementation of certain models, whereas SpaCy will be employed for preprocessing scripts, facilitating lexical modifications in texts and enabling result comparisons.

In addition to SpaCy, we also employed the Gensim library for obtaining semantic similarities of words within the text processing domain. Furthermore, for certain tests, we utilized the HuggingFace Transformers' *pipelines*, using the PyTorch implementation backbone. These pipelines have proven to be the most efficient and fast approach for conducting inference tests, particularly for NLP tasks.

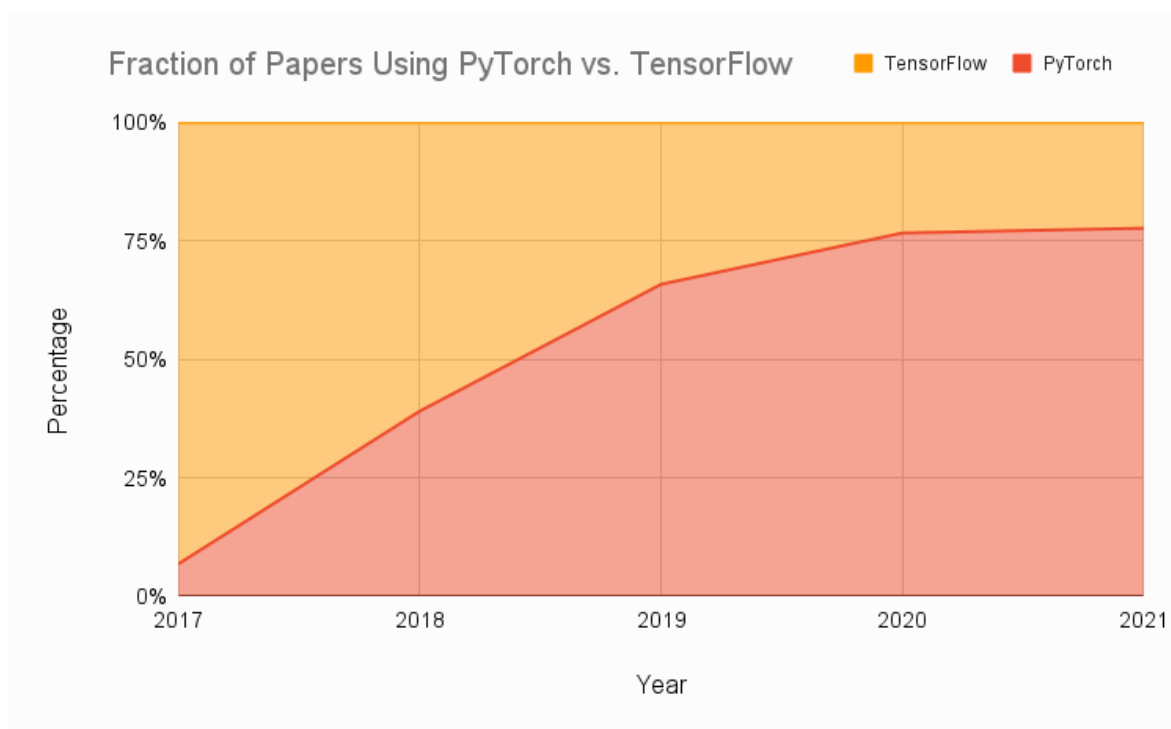


Figure 3.2: Popularity comparison of TensorFlow and PyTorch inside the research community.

4. Proposed architecture

After selecting the technologies, software, and hardware as the platform for our experimentation, our next step is to create the necessary components to achieve our goals. This chapter will primarily focus on presenting the components used in various tests conducted throughout the experimentation process. In Section 4.1, we will introduce the datasets employed during the experimentation. Then, in Section 4.2, we will discuss the different components that will form the proposed pipelines, which will be explored further in the next chapter. Finally, in Section 4.3, we will provide explanations of the diverse evaluation methods utilized to test the effectiveness of the proposed pipelines.

4.1. Datasets

In this section, we will provide a list of the datasets used as input for the various experimental tests conducted. Additionally, we will delve into the data and labeling provided by these datasets, analyzing their suitability for our specific purposes.

4.1.1. Requirements

For the experimentation we plan to conduct, we require different datasets that possess specific characteristics. Firstly, to evaluate the proposed architectures, we need datasets comprising video scenes captured in diverse home environments. It is important that these datasets contain annotations with detailed descriptions illustrating the events depicted in the videos. These annotations should facilitate the evaluation of the Video Captioning pipeline. Additionally, the annotations should provide information about the most relevant objects appearing in the videos to enable testing of the Object Recognition pipeline. Secondly, we require a dataset that presents a collection of risks associated with objects commonly found in houses. This dataset will be used for conducting risk assessments.

As we have previously mentioned in earlier chapters, datasets that meet our specific requirements are quite limited. Therefore, we have adopted a flexible approach in selecting datasets. We have not only chosen datasets that fully meet all the requirements but also selected datasets that we believe have the potential to yield satisfactory results doing some modifications to the original proposed pipelines.

4.1.2. Selection

The first dataset we required was for the computation of the description and Object Recognition pipelines. To fulfill this requirement, we selected the Charades and ETRI-Activity3D datasets as references for our experimentation.

The Charades dataset, which was explained in Section 2.5.3, consists of 9,848 videos showcasing daily home activities performed by 267 individuals from three different countries. This

ensures visual diversity and richness. The dataset’s annotations provide valuable information about the scenes, objects, and actions occurring in the videos. Additionally, it offers multiple descriptions generated by different annotators for almost all the videos, with each video having at least one annotated description. This dataset will be the primary dataset used during our experimentation as it meets all the requirements outlined in the previous subsection. We also selected the ETRI-Activity3D dataset, which was explained in Section 2.5.4, as a secondary dataset. This dataset contains videos of home environments featuring three different scenes, where 55 distinct actions are performed by a group of 100 actors spanning two different age ranges. The dataset’s labeling includes information about the actions, actors, scenes, and the number of cameras used to record the videos. However, in our experimentation, we will only utilize the action annotations. We chose this dataset due to its visual content alignment with our requirements, even though the provided annotations were not directly useful for our purposes. In Table 4.1 is shown a summary of the data provided by the previously mentioned and explained Charades and ETRI-Activity3D datasets.

Features	Charades	ETRI-Activity3D
Number of videos	9,848	112,620
Dataset purpose	Action recognition	Action recognition
Application	General context home Action recognition	Home Action recognition for elder care robots
Data modalities	RGB videos	RGB videos, depth map frames, body index frames, 3D skeletal data
Subjects	267 (all age ranges)	100 (50 in 64-88 and 50 in their 20s)
Annotated descriptions	Yes (one or more)	No
Annotated objects	Yes	No
Annotated actions	Yes	Yes

Table 4.1: Summary table of data provided by Charades and ETRI-Activity3D datasets.

Our second requirement was related to risk assessment, where we needed a dataset that could establish a relationship between detected objects and potential risks. Unfortunately, we were unable to find suitable existing datasets for this purpose. Consequently, we decided to create our own corpus by referencing related datasets. The creation of the corpus involved three core tasks: collecting a set of common objects found in homes, collecting a set of risks associated with a home environment, and establishing the relationship between the objects and risks. The next subsection will provide a detailed explanation of the taxonomy and reference datasets used in the creation of the corpus.

4.1.3. Risks-objects corpus

Each corpus is structured according to a taxonomy. In our proposed corpus, the primary item is the object. Each object is composed of three different fields: the object’s name, a list of specific risks associated with the object, and a list of properties defining common risks applicable to more than one object in the corpus. Each property consists of a name and a list of risks, while each risk is defined by an identifying name, an English description, and the corresponding translation into Spanish.

The corpus is stored in JSON file format and contains a list of 56 distinct objects. In Table 4.2 are shown a summary of the 56 different object comprising the entire corpus. We define an ”object” as any item commonly found in homes, ranging from objects like chairs to food items like apples or pets like dogs.

Objects			
Ball	Bicycle	Bench	Bird
Cat	Dog	Backpack	Umbrella
Handbag	Tie	Suitcase	Frisbee
Baseball bat	Racket	Skateboard	Bottle
Wine glass	Cup	Fork	Knife
Spoon	Bowl	Banana	Apple
Sandwich	Orange	Broccoli	Carrot
Hotdog	Pizza	Donut	Cake
Chair	Couch	Plant	Bed
Table	Toilet	TV	Laptop
Mouse	Remote	Keyboard	Cellphone
Microwave	Oven	Toaster	Sink
Refrigerator	Book	Clock	Vase
Scissors	Teddy bear	Hairdryer	Toothbrush

Table 4.2: Complete set of 56 objects that compose our Risks-objects corpus used during experimentation.

Previously, we outlined three tasks to construct the corpus: collecting a set of objects, a set of risks, and associating the objects with the corresponding risks. For collecting objects, we used the COCO2017¹ (T.-Y. Lin et al., 2015) dataset as a reference for technical reasons. As we will discuss in the following section, we will utilize a YOLO model pretrained on COCO2017 as the core for Object Recognition, thus restricting the detection to objects

¹Reference to the dataset version of 2014 was cited as 2017 version has no associated official publication.

```

1 {
2   "name": "cup",
3   "properties": [fragile, liquid_container]
4   "specific_risks": []
5 }

```

Figure 4.1: Example of object included in the Risks-objects corpus. The cup object will acquire the risk *breakable* associated with the property *fragile* and the ones associated with the property *liquid_container*. Objects can have no specific risks.

```

1 {
2   "name": "fragile",
3   "risks": [
4     {
5       "name": "dangerous_pieces",
6       "en": "If it is made of a fragile material and collides with something, it can ↔
7         ↔ shatter into dangerous pieces that can cause injuries.",
8       "es": "Si está hecho de un material frágil y choca contra algo, puede romperse ↔
9         ↔ en pedazos peligrosos que pueden causar lesiones."
10    }
11  ]
12 }

```

Figure 4.2: Example of property included in the Risks-objects corpus.

within this dataset. To collect risks, we created new risks based on the objects provided by the COCO2017 dataset. We analyzed each object in the dataset and identified potential risks associated with them. Finally, to associate risks with objects, we utilized specific risks and properties. Specific risks are exclusive to the object being defined, while properties represent object features that define possible risks based on essential characteristics such as manufacturing materials or the ability to store fluids. When a property is associated with an object, the object inherits the risks defined by that property. The concept of a *property* used here was extracted from the objects-actions corpus of the VirtualHome simulator, where properties defined potential actions that could be performed on an object at a specific moment in the simulation. Figures 4.2 and 4.1 show examples of a property and an object from the proposed corpus. The complete corpus of properties and objects is shown in Appendix B.

4.2. Modules

In this section, we will describe the main components responsible for conducting the experimentation. These modules encompass various aspects, from DL models to data processing components that transform data prior to evaluation.

4.2.1. Video captioning module

The Video Captioning module is the cornerstone of our framework, as it generates natural language descriptions from visual data. In this section, we will provide details about the Video Captioning model used to obtain these descriptions from input videos.

For this module, we utilized the SwinBERT Video Captioning model (K. Lin et al., 2022). Developed by Microsoft Open Source in 2022, the SwinBERT model aims to create an end-to-end solution for the Video Captioning task. It is an end-to-end model based on Transformers, introducing notable advancements such as adaptable spatial-temporal feature encoding for variable-length frames and a trainable sparse attention mask. This attention mask allows the model to focus on frames with more significant spatial-temporal movements, thereby optimizing task-specific performance. SwinBERT is one of the most prominent Video Captioning models available today, demonstrating substantial improvements over previous methods on widely-used Video Captioning datasets. In our experimentation, we will employ this model with various pretrainings provided in its scientific publication. Figure 4.3 illustrates the internal architecture of the SwinBERT model. It follows the trend of Video Captioning models that adopt a two-stage encoder-decoder approach.

The first stage of the Video Captioning module utilizes the Video Swin Transformer (Liu et al., 2021) as encoder. This model achieves a favorable speed-accuracy trade-off while maintaining densely sampled video frames. It takes raw video frames as input and produces video spatial-temporal feature tokens. The VidSwin model is pretrained on the Kinetics action recognition task.

In the second stage, we employ as decoder a Transformer to generate natural language descriptions. The input to this stage includes the feature tokens extracted in the first stage, as well as word tokens. The output is obtained through a seq2seq generation process, where word tokens have self-attention masks that restrict their attention only to existing output tokens. Additionally, there is full attention to the feature tokens from the first stage.

Another important component within the model architecture is the Sparse Attention Mask. It plays a vital role in the overall structure. This attention mask acts as a regularizer for the multimodal transformer decoder (second stage), reducing redundancy among the different video tokens. Due to the dense sampling of frames, the tokens can be redundant, which may impact performance. The mask focuses on the active video tokens that contain rich spatial-temporal information. In this model, a sigmoid activation is applied to the mask, ensuring values are distributed between 0 and 1.

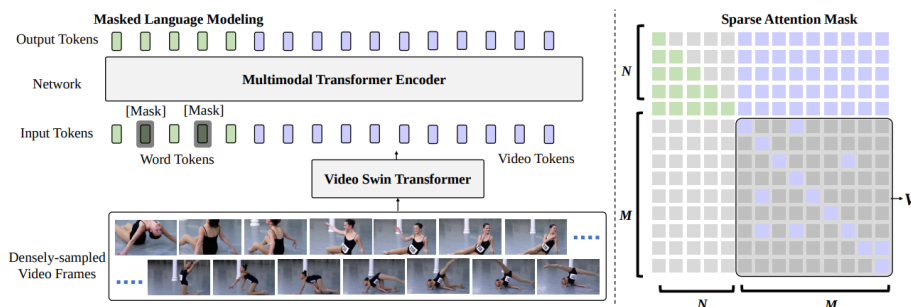


Figure 4.3: SwinBERT internal architecture.

4.2.2. Pre-processing module

In this module, our focus is on manipulating the lexical aspects of the descriptions prior to evaluation. We apply lexical transformations to the data that will be evaluated to demonstrate how lexical information impacts metric evaluation.

In captioning tasks, it is not only important to compare individual words, n-grams, or subsequences of tokens, but also to capture the underlying semantics of the compared sentences. To maximize the extraction of semantic information, we conducted experiments using various preprocessing techniques on the sentences before comparison.

We performed different combinations of preprocessing approaches to compare their effects on the lexical results. The individual operations applied are summarized as follows:

- **Lemmatization:** It consists of obtaining the root lemma of each of the tokens. This mainly helps us to avoid the rich lexical diversity, such as verb tenses.
- **POS Filtering:** Part-of-speech (POS) tagging consists of assigning to each token a tag that represents its type of word. Once the tokens are tagged, we filter them to only capture verbs, nouns, proper nouns, and adjectives because these types of words constitute the semantic meaning of a sentence.
- **Punctuation removal:** As we detected that punctuation signs have a considerable impact on the results of some samples, we tested the application of a punctuation removal step.

Figure 4.4 demonstrates the application of the aforementioned transformations on a sample sentence. To obtain all the lexical information about each token in the analyzed set of sentences, we utilized the spaCy library².

S = A man is running, he looks tired.

Lemma(S) = A man be run, he look tire.
POS(S) = man running, looks tired.
Punct(S) = A man is running he looks tired

Figure 4.4: Applying individual transformation over a sample sentence.

4.2.3. Object extraction module

Once we have generated descriptions using SwinBERT, the next step is to extract the objects mentioned in these captions. Although we will use SwinBERT checkpoints pretrained on large-scale datasets that can recognize a wide range of objects, we will limit the variety of objects to 56 different classes due to temporal restrictions established for developing this thesis. The Object Recognition module, which is based on a backbone model pretrained on a limited set of 80 classes, can only detect 56 items which are commonly found in homes. The details of the Object Recognition module will be explained in the next subsection.

²<https://spacy.io/>

To extract objects from the captions, we perform element-wise operations where each token in the captions is compared with each possible class in our object set. During this comparison, we need to consider the underlying semantics of the tokens, as different tokens with different lexicons can refer to the same concept. To address this, we utilize the Gensim library to obtain vector embeddings representing the words. Instead of directly comparing the token lexicons, we compare these embeddings. The comparison of embeddings is computed using cosine similarity, which measures the distance between the embeddings. We used a threshold of 0,62 to determine if a pair of tokens is a match, this means that the cosine similarity between the embeddings must be equal to or higher than 0,62 for the SwinBERT token to be considered a match with the object class being compared.

To determine the specified threshold, we compared the embeddings of our object set with the embeddings of typical words used to mention objects commonly seen in homes. For example, we compared "tv" with "television" and "sofa" with "couch", this allowed us to understand the cosine similarity value between words that refer to the same object. Additionally, we performed the same comparison using embeddings of words that are related but do not represent the same objects. For instance, we compared "refrigerator" with "oven" (both furniture) and "apple" with "banana" (both fruits), this helped us obtain the typical threshold for words that are related but do not represent the same object. After computing these comparisons across multiple examples, we observed that the lowest cosine similarity value for objects referring to the same object was obtained between "tv" and "television," resulting in a value of 0,6426. On the other hand, the highest cosine similarity value for words referring to different objects was obtained between "refrigerator" and "oven," yielding a value of 0,6128. Therefore, we determined that 0,62 is the optimal threshold for distinguishing between correct and incorrect matches based on our proposed COCO2017 classes subset.

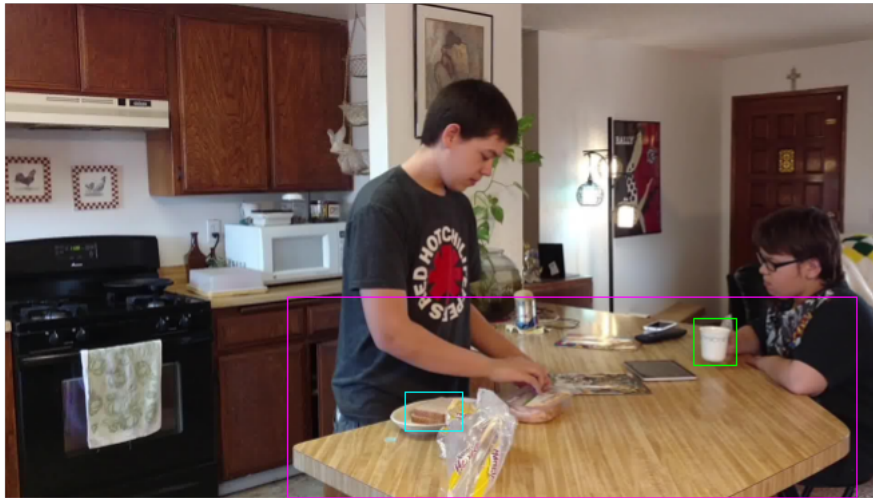
To obtain the vector embeddings, we used the pretrained model *word2vec-google-news-300*, which is available through the Gensim API. In Figure 4.5 is shown an execution sample of this module.

4.2.4. Object recognition module

The Object Recognition module serves as an auxiliary component in the object detection stage of the architecture. It complements SwinBERT helping in the prioritization of objects importance and enhancing video comprehension. This module consists of a fully-aimed Object Recognition model that captures more detailed information about the presence of objects compared to SwinBERT.

As a reference for this module, we have chosen the YOLOv7 model (C.-Y. Wang et al., 2022). YOLOv7 is the latest official release from the YOLO family of models, designed for real-time Object Recognition in videos and images. It is currently considered the fastest and most accurate real-time object detection model for computer vision tasks. YOLOv7's architecture is based on previous models from the YOLO family, such as YOLOR (C.-Y. Wang et al., 2021), and its key features and advancements can be summarized as follows:

- **E-ELAN:** The Extended Efficient Layer Aggregation Network is the main component of the model's backbone. It includes the "expand, shuffle, merge cardinality" operation,
-



Description: There is a person sitting at the *table* drinking from a *glass*. There is another person standing at the *table* making a *sandwich*.

Objects: Table, glass, sandwich

Figure 4.5: Execution sample of the object extraction module over Charades input video.

which allows the model to continuously learn without destroying the original gradient path. This enables the model to learn more diverse features.

- **Compound model scaling:** This feature maintains the initial design of the network while allowing scaling of any attribute of the model. Models can scale key attributes to meet the requirements of different applications. With compound scaling, the models maintain an optimal structure when scaling attributes.
- **Planned re-parameterized convolution:** RepConv (Ding et al., 2021) is a CNN architecture characterized by using a structural re-parameterization technique that enables the model to have a different inference and training body. YOLOv7 includes RepConv without identity connections.
- **Multi-head:** YOLO models consist of a backbone, a neck, and a head. In YOLOv7, the architecture consists of two different heads: the lead head, responsible for obtaining the final outputs, and the auxiliary head, which assists in training the middle layers.
- **New label assignment:** YOLOv7 incorporates a new label assignment strategy. The network considers prediction results along with the ground-truth and assigns soft labels instead of the traditional approach where label assignment directly refers ground-truth to assign hard labels based on given rules.

During the experimentation, we will utilize the YOLOv7 model pretrained on the COCO2017 dataset, which enables the YOLOv7 model to recognize up to 80 different object classes. However, we will focus on a subset of 56 objects that are relevant to home scenes. Due to time

constraints we have not conducted any training stages or tests using other pretraining checkpoints. Future studies will consider expanding the number of classes that can be recognized by the model. Additionally, we will test YOLOv7 using a confidence threshold of 0,7. This means that the model will only output detected objects in which it has at least 70% confidence in its prediction.

For the selection of the aforementioned threshold, we analyzed the confidence values of object predictions from multiple videos. Our goal was to identify the range of confidence values at which our YOLOv7 model tends to avoid outputting hallucinations. Through this analysis, we discovered that a threshold of 0,7 results in correct detections in the majority of cases. While this threshold is effective in preventing critical hallucinations, it may not completely eliminate confusion in cases involving objects with similar features, such as "remote" and "cellphone". However, this is not a major concern since, as we will show in subsequent sections, this module will be used to construct a Risk Assessment pipeline in which similar objects will have similar output risks. We also recognize the importance of maintaining flexibility with the Object Recognition threshold to avoid overlooking important risks. In Figure 4.6 is shown an execution sample of YOLOv7 model over a Charades video.



Figure 4.6: Execution sample of the YOLOv7 model over a Charades input video.

4.2.5. Risks matching module

The risks matching module is responsible for assessing risks based on the objects detected in the video. It represents the final step in the risk assessment pipeline. This module takes two inputs: the objects recognized by SwinBERT and the objects detected by YOLOv7. It outputs the potential risks identified in the scene based on the input objects. Additionally, it provides a risk level associated with each detected risk, which is based on the relevance of an object within the video.

The module consists of two main core parts. First, the objects are prioritized by relevance. Then, the objects and risks are matched using the risks-objects corpus described in Section 4.1. To compute the relevance prioritization of objects, we use three different levels. Level

1 represents the lowest relevance and includes objects detected only by the YOLOv7 model. Level 2 includes objects detected only by the SwinBERT model. Finally, Level 3 represents the highest relevance and includes objects detected by both the SwinBERT and YOLOv7 models. The relevance mechanism prioritizes objects detected by SwinBERT because this model tends to capture the most relevant parts of scenes, while YOLOv7 aims to detect as many objects as possible. Once the objects are prioritized by relevance, we compute the risks matching process. This process involves searching for each detected object within the corpus and extracting the associated risks. During risk extraction, we retrieve not only specific risks but also risks associated with the properties defined for the analyzed object. Execution samples of this module will be shown at the end of Section 5.3, where we test the complete Risks Assessment pipeline over different input videos.

4.3. Evaluation

Evaluation metrics play a crucial role in the complete pipeline, as it is essential to ensure that the metrics and methods used align with the task being performed. In this section, we will introduce the set of evaluation methods that will be employed during the experimentation.

4.3.1. BLEU Score

BLEU (Papineni et al., 2002) is one of the most well-known metrics in the NLP community and is commonly used in tasks such as text translation. It measures the quality of a translation by comparing translated sentences to reference sentences. BLEU calculates the number of matching n-grams between the references and translations and produces a score between 0 and 1, with 1 being the highest score.

There are several variants of BLEU based on the value of n for n-grams. The most commonly used variants are BLEU-1, BLEU-2, BLEU-3, and BLEU-4, with BLEU-4 being the most stringent score. An example of different BLEU variants applied to a reference-candidate sample can be seen in Figure 4.7.

While BLEU is widely used in NLP tasks, it has certain limitations that should be considered. One significant limitation is that BLEU only compares n-grams and does not take into account the underlying semantics of the lexical information. An example of this is the pair of words "chair" and "seat." While these words may be considered distinct from a lexical perspective, they share the same semantic meaning as they both refer to the same type of object. Another of its limitations is also related with its lexical focus, as it computes direct matches, sometimes assigns higher scores to sentences that use similar words (such as prepositions or articles) regardless of their actual quality. Despite these limitations, we will use this metric due to its importance in the NLP community. However, we will also compute other metrics that are more suitable for our task. For evaluating BLEU score, we have chosen the SacreBLEU implementation from TorchMetrics³. We opted for BLEU-1 score because it is the least lexically restrictive metric, as it compares unigram matches between sentences.

³<https://torchmetrics.readthedocs.io/en/latest/>

Reference = A boy is playing football with his friends
Candidate = A group of kids are playing football

BLEU-1 = 0.37
BLEU-2 = 0.23
BLEU-3 = 0.0

Figure 4.7: Comparison between variants of BLEU score.

4.3.2. BERT Score

BERT Score (Zhang et al., 2020) is a metric that was developed to address the limitations of BLEU and to consider the semantic information in text evaluation. It compares the similarity between two texts using a pretrained Bidirectional Encoder Representations from Transformers (BERT) model (Devlin et al., 2019). BERT is a context-aware language model based on Transformers that generates word embeddings for various NLP tasks. Word embeddings are vector representations that capture the meaning of a text.

In BERT Score, the metric utilizes BERT to obtain word embeddings for the tokens in the compared sentences and then calculates the cosine similarity between the embeddings. The output of this comparison consists of three values: Precision, Recall, and F1 score. Each of these values ranges from 0 to 1, with 1 indicating the highest score.

By using context-aware embeddings, BERT Score takes into account both the lexical and semantic information for evaluating the similarity between texts. This metric has gained popularity for evaluating text generation systems because it can capture semantic nuances and is less sensitive to text length variations compared to BLEU. In Figure 4.8, an example of BERT Score evaluation is shown, where the sentence with the highest semantic similarity to the reference sentence receives the highest score.

To compute BERT Score, we utilized the official implementation available on their GitHub repository⁴ with the setting “roberta-large.L17.no-idf.version=0.3.12(hug.trans=4.28.0.dev0)-rescaled.fast-tokenizer”. This setting employs the *roberta-large* model as the BERT backbone architecture, specifically using the output from layer number 17. It is important to note that this setting includes rescaling of the output from *roberta-large* using a predefined baseline to adapt the score range to 0-1. This rescaling is necessary because BERT Score calculates cosine similarity, and the default output range of cosine similarity is -1 to 1.

4.3.3. Zero-shot classification

The Zero-shot classification task involves classifying a sentence into a specific class. In this task, we provide a sentence and a set of labels to the model, and the model’s objective is to determine how well the labels align with the meaning of the sentence. This task is typically performed using large pre-trained models, as the effectiveness of the results heavily relies on the amount of information the model has learned during training.

Although Zero-shot classification is not strictly an evaluation metric, we will utilize it as an

⁴https://github.com/Tiiiger/bert_score

Reference = He is going to travel to New York
Candidate 1 (C1) = He wants to go camping
Candidate 2 (C2) = He wants to visit America

F1 (C1) = 0.4826
F1 (C2) = 0.6181

Figure 4.8: Example of BERT Score evaluation using F1. The candidate with highest semantic similarity achieves the highest score.

Sentence = Dune is the best movie ever.
Labels = [cinema, art, music]

Score(cinema) = 0.90
Score(art) = 0.10
Score(music) = 0.00

Figure 4.9: Example of Zero-shot classification determining scores of a sentence given 3 possible labels.

evaluation method following the procedure outlined below. We will feed a natural language Video Captioning description to a Zero-shot classification component and test whether it can predict the annotated class that corresponds to the input video description. We will then calculate the accuracy of the predictions to assess the performance of this evaluation. To implement this, we will leverage the HuggingFace Transformers library’s *pipelines* functionality and utilize the *bart-large-mnli*⁵ model as the backbone for label scoring. In Figure 4.9 an example of the Zero-shot classification Transformers *pipeline* is shown.

4.3.4. Object Recognition Recall Score

Within the context of Object Recognition, one commonly used metric for evaluation is Recall. Recall measures how effectively True Positives (TP) are identified among all predictions, which is the sum of TP and False Negatives (FN). The formula for calculating Recall can be seen in Figure 4.10. Recall serves as the foundation for more complex metrics like Mean Average Precision (mAP), which is a widely known metric in object detection. However, we will not be using mAP for evaluation since our datasets do not provide ground-truth bounding box annotations, which are required for computing this metric.

The traditional Recall metric relies on Intersection over Union (IoU) to distinguish between TP and False Positives (FP). IoU is another Object Recognition metric that indicates the overlap between ground-truth bounding boxes and predicted bounding boxes. Recall uses an IoU threshold to determine if a predicted object is a TP (if the prediction confidence is higher than the threshold) or an FP (if the prediction confidence is lower than the threshold). Since our datasets lack ground-truth bounding box annotations, we are unable to compute the IoU

⁵<https://huggingface.co/facebook/bart-large-mnli>

$$Recall = \frac{TP}{TP + FN} \quad (4.1)$$

Figure 4.10: Recall formula.

metric. Therefore, we will make slight modifications to the classical approach of the Recall metric to adapt its calculation to our specific requirements.

As mentioned in Section 4.2.4, we are constrained to detect a limited number of classes. Therefore, we will filter the ground-truth annotations by removing any objects that are not part of the set we can detect. After this filtering process, we will calculate the Recall metric by considering all the detected objects as True Positives (TP). Objects from the ground-truth set that were not detected will be considered False Negatives (FN).

5. Experiments and results

Once we have reviewed the different components, modules, and evaluation methods that will be utilized during the experimentation phase, we will delve into the experimental pipelines proposed to achieve the goals which motivated this study. For each of the proposed pipelines, we will provide an explanatory diagram and showcase the results obtained from the tests. Since this study involves the interaction of various components, it is crucial to test the models and datasets under different conditions to demonstrate both their effectiveness and limitations. Consequently, not all the proposed pipelines align with the primary objective of this study, and evaluation pipelines have also been included. The purpose of each pipeline will be clearly explained. All the implemented code used to carry out the entire experimentation developed is available in the GitHub repository of the thesis¹.

In Section 5.1, we present the pipelines associated with the evaluation of Video Captioning architectures. Following that, in Section 5.2, we introduce the pipeline designed for evaluating the Object Recognition architecture. Finally, in Section 5.3, we present the pipeline dedicated to Risk Assessment.

5.1. Descriptions evaluation

In this section, we will delve into the experimentation related to description generation. Here, we will examine the proposed pipelines for evaluating the captions generated by the Video Captioning module. The main evaluation metrics we will use are BLEU-1 and BERT Score. These two metrics have been selected because they offer different evaluation perspectives. BLEU-1 is a commonly used metric in NLP that primarily assesses the lexical similarity between descriptions. On the other hand, BERT Score is a more recent metric that evaluates descriptions from a semantic standpoint as well. In addition, we will also introduce the Zero-shot classification task as an evaluation method. This allows us to leverage datasets that do not provide annotated descriptions but instead provide annotated classes for actions.

5.1.1. Zero-shot classification-based pipeline

In this first proposal, we will analyze the implementation of the Zero-shot classification task as an evaluation method. As mentioned earlier, the Zero-shot classification task involves classifying sentences based on a given list of input labels. This task assigns a score to each label, indicating the degree to which the label aligns with the underlying semantics of the provided sentence.

For evaluating generated texts in Video Captioning, annotated descriptions provided by the dataset are typically used. However, there are certain datasets, particularly those focused on activity recognition in home environments, where the annotations are not descriptions but

¹<https://github.com/javirodriguez/indoor-risks-assessment>

rather classes. In our study, we are specifically interested in datasets that include videos of individuals performing household and daily tasks, which often have class-based annotations. One such dataset is ETRI-Activity3D, which we will use for this architecture. We have two main goals for this experiment: Firstly, we aim to demonstrate the capability of SwinBERT to generate accurate descriptions in indoor environments. Secondly, we want to test if we can leverage datasets with class-based annotations for our purposes.

In this section, the proposed architecture will use the first 880 videos from the ETRI-Activity3D dataset as testset. We have selected this subset of videos because it encompasses all possible scenes and actions available in the dataset, performed by the same person. This subset will provide sufficient material for the experimentation we intend to conduct. The remaining portion of the dataset includes the same scenes and actions but performed by different subjects. We will employ the Video Captioning model SwinBERT, pretrained on the MSVD D. Chen & Dolan (2011) dataset, to generate natural language descriptions from the videos. MSVD is a popular large-scale dataset for Video Captioning task comprised by more than 2,000 videos annotated with an approximate total sum of 120,000 descriptions with a maximum length of one line. This dataset was created through Amazon Mechanical Turk. We selected MSVD for pretraining because it provides shorter descriptions that are more likely to be similar to the annotated classes, thus facilitating the labeling process. The test set comprises videos covering all possible actions within the dataset, performed by the same person, and recorded from different perspectives in a minimum of 2 and a maximum of 4 different rooms, depending on the action. Each video is identified by a series of characters, with a portion of the identifier representing the action class being performed. We will utilize this information in the evaluation step.

To compute the evaluation, we will employ a Zero-shot classification pipeline from the Transformers library, using as backbone the model *bart-large-mnli*. We will provide the pipeline with a SwinBERT description and each possible label from the ETRI dataset, where each label represents an action. Finally, we will compare the predicted classes with the annotations in the video identifiers and calculate the Recall of predictions for the test set. We will only provide Recall score as we are only interested in knowing the percentage of correct predictions from all of them. Moreover, this will allow us to accelerate the development of the experiment which is important due to the temporal limitations mentioned in previous sections. Figure 5.1 depicts the complete pipeline, with the evaluation stage components highlighted in red.

After conducting the evaluation, we found that this pipeline correctly predicts the label for at least one scene of the action in approximately **34.55%** of the actions. This represents a correct recognition of 19 actions out of a total of 55. Additionally, we observed that when the model accurately classifies an action, it tends to classify more videos correctly where that action is being performed. The Recall percentages obtained for each action can be found in Appendix I, specifically in Tables A.1 and A.2. These percentages were calculated by dividing the number of correctly labeled videos for an action by the total number of videos for that action.

Although the results obtained are not as high as desired, a careful examination reveals that most of the errors occur during the labeling process rather than in the retrieval of descriptions. It is important to consider that we are attempting to classify an interpretation

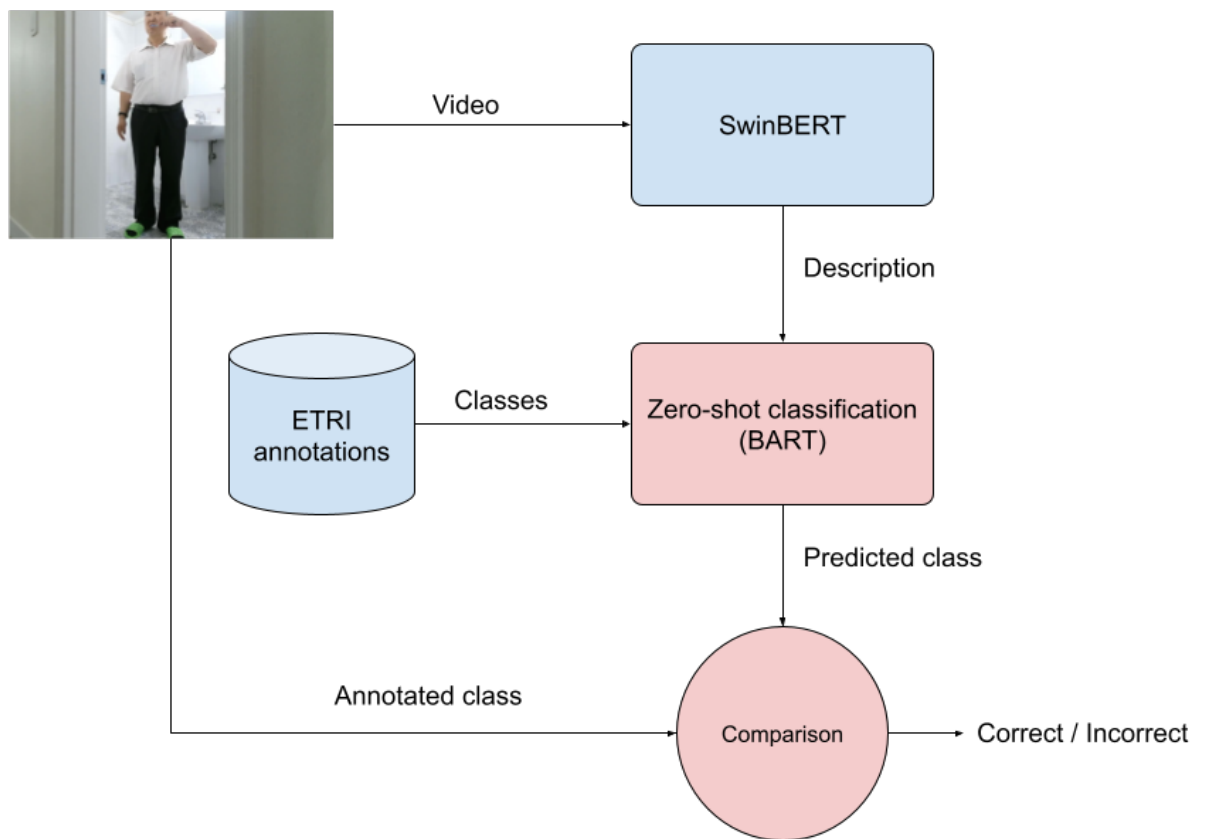


Figure 5.1: Zero-shot classification of SwinBERT descriptions experiment pipeline.



Description: A man is opening a freezer
Predicted class: pointing with a finger
Annotated class: putting food in the fridge/taking from the fridge

Figure 5.2: Sample of wrong labeling given a correct description.

of a video into a fixed class. Thus, it is possible that while the description of the video is not incorrect from a human interpretation perspective, the assigned class may refer to a different event occurring in the video. Examples of these issues can be seen in Figures 5.2 and 5.3. Furthermore, we must take into account that we are performing a zero-shot classification task on a set of 55 different classes. Due to the large size of the class set compared to a typical zero-shot classification task, it is easy for the BART model used to confuse classes. These factors explain the relatively low results obtained in this test.

Further studies on zero-shot classification are required to fully assess the feasibility of using datasets with class-based annotations. However, these studies will be considered for future works as we are restricted to one academic year for the composition of this thesis.

5.1.2. Metric-based evaluation pipeline

In this second proposal, we will focus our experimentation on evaluating the architecture using established metrics commonly used in the field of NLP. Since we will be working with text evaluation metrics, we will also leverage the textual pre-processing module to demonstrate how lexical changes can impact the obtained results.

In this section, our objective is to evaluate the natural language descriptions generated by SwinBERT from the videos of the Charades dataset, utilizing state-of-the-art metrics for result computation. Unlike the ETRI-Activity3D dataset, Charades provides richer annotation data, including descriptions that explain the events in the videos. Therefore, we focus on the Charades dataset as it contains the exact type of annotations required for our evaluation.

We conducted tests on a randomly selected set of 3,000 videos from the Charades dataset. These videos were processed using three different SwinBERT pretrainings: MSVD, MSR-



Description: a man is standing in a kitchen
Predicted class: sitting up/standing up
Annotated class: drinking water

Figure 5.3: Sample of wrong labeling because of unexpected description.

VTT (Xu et al., 2016), and VATEX (X. Wang et al., 2020). For each pretraining, we performed various experiments by applying different textual preprocessing techniques to maximize semantic similarity. The individual textual preprocessing techniques tested were: Raw (no transformation applied), POS (part-of-speech filtering, retaining only nouns, proper nouns, adjectives, and verbs from the complete descriptions), Lemma (lemmatization of tokens), and Punct (removal of punctuation). We employed the different preprocessing techniques and combined them in all possible ways to showcase results from different perspectives. Additionally, it is worth mentioning that the VideoSwin model used during the encoding phase of SwinBERT was initialized with pretrained weights from Kinetics-600 (Carreira et al., 2018).

The evaluation of the results utilized two metrics: BLEU-1 and BERT Score. As mentioned earlier, BLEU-1 measures the matching of n-grams between the generated description and the target annotation. On the other hand, BERT Score obtains contextual embeddings for each token in the descriptions using a BERT model, and then compares these embeddings to calculate precision, recall, and F1 scores. Unlike BLEU, which solely relies on lexical comparisons, BERT Score can compare sentences at a semantic level, assigning high similarity scores to different words that are synonymous. Figure 5.4 illustrates the complete pipeline for this experiment. Similar to the pipeline described in Subsection 5.1.1, the evaluation stage components are highlighted in red in this figure.

Table 5.1 presents the results of the experiments conducted on a randomly selected sample of 3,000 videos from the complete Charades dataset, measured in terms of the BLEU-1 metric. The highest score in each table is highlighted in bold. Furthermore, Figure 5.5 provides an example showcasing the different SwinBERT descriptions obtained using the various pretrainings on the same video.

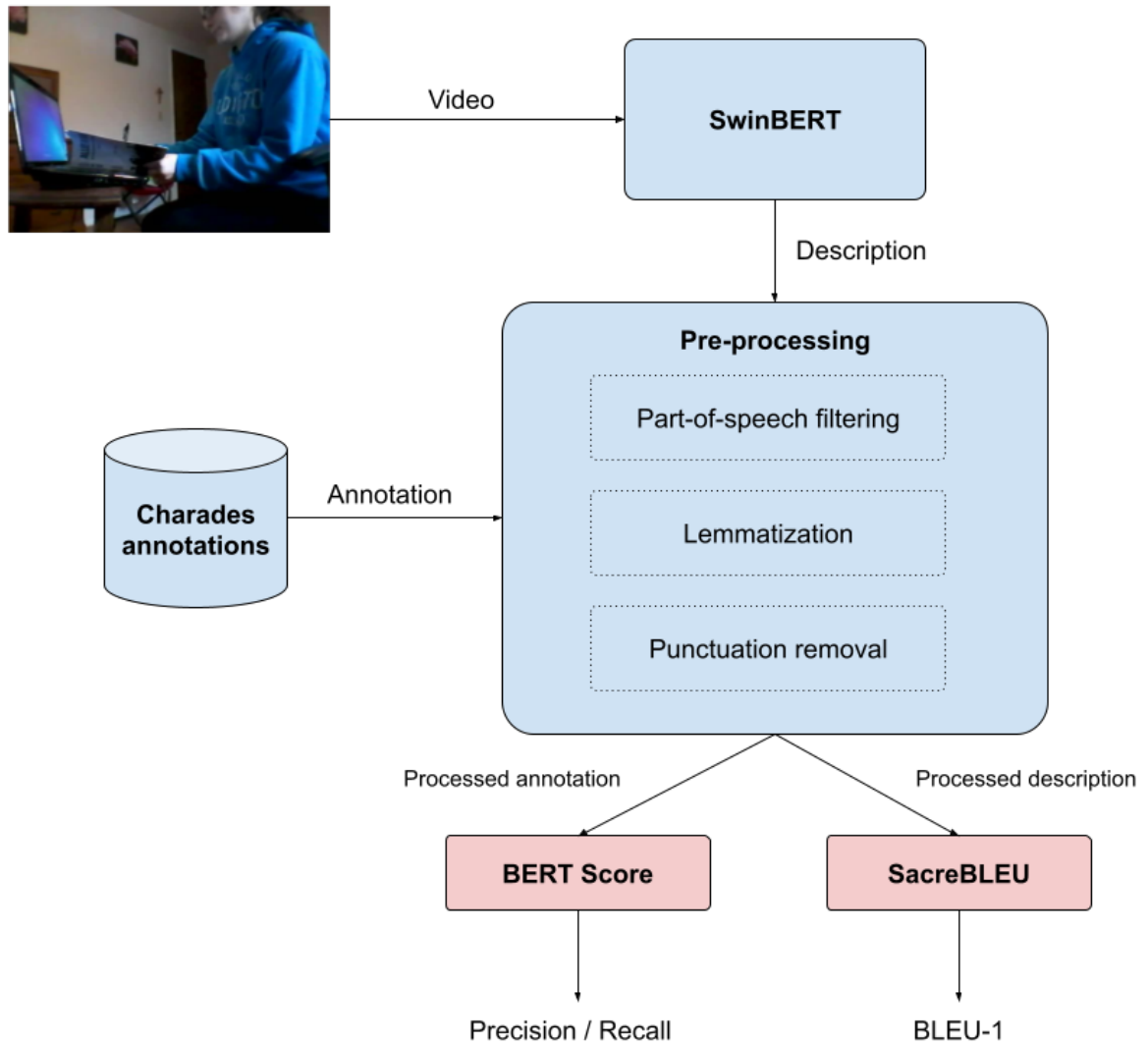


Figure 5.4: BLEU-1 and BERT Score evaluation of SwinBERT descriptions experiment pipeline.



GT: A person wrapped in a blanket is sitting at a table and eating something from a bowl. Lastly the person takes a drink from a cup.
MSVD: a man is drinking a drink from a cup
MSR-VTT: a man is eating something in his kitchen
VATEX: a young man is sitting at a table and drinking from a cup.

Figure 5.5: Charades' frame example with its ground-truth (GT) annotation and three different captions obtained from SwinBERT (MSVD, MSR-VTT and VATEX).

Pre-processing	MSVD	MSR-VTT	VATEX
Raw	0.0932	0.1207	0.2864
POS	0.0349	0.0339	0.1828
Lemma	0.1110	0.1268	0.3314
Punct	0.1146	0.1307	0.2921
Punct+POS	0.0418	0.0408	0.1635
Punct+Lemma	0.1211	0.1374	0.3070
Lemma+POS	0.0436	0.0423	0.2057
Punct+Lemma+POS	0.0525	0.0511	0.1892

Table 5.1: BLEU-1 score results of testing a sample of 3,000 videos with different textual pre-processing methods.

Pre-processing	MSVD		MSR-VTT		VATEX	
	Precision	Recall	Precision	Recall	Precision	Recall
Raw	0.454	0.198	0.434	0.199	0.454	0.310
POS	0.258	0.104	0.225	0.088	0.326	0.179
Lemma	0.328	0.128	0.313	0.137	0.366	0.266
Punct	0.467	0.166	0.447	0.166	0.424	0.270
Punct+POS	0.244	-0.018	0.214	-0.034	0.217	0.071
Punct+Lemma	0.329	0.085	0.315	0.095	0.322	0.217
Lemma+POS	0.223	0.090	0.176	0.068	0.307	0.171
Punct+Lemma+POS	0.208	-0.029	0.171	-0.045	0.217	0.087

Table 5.2: BERT Score results (precision and recall) of testing a sample of 3,000 videos with different textual preprocessings.

In Table 5.1, we can observe the differences obtained by using one pretraining or another, with the best results obtained using the VATEX checkpoint. VATEX had higher BLEU-1 scores because this SwinBERT checkpoint is capable of obtaining more detailed and larger descriptions compared to MSVD and MSR-VTT. This contributes to more n-gram matches as the descriptions have a more similar length to the annotated descriptions. Moreover, we observed that removing tokens from the descriptions decreased the scores, which can be explained by the nature of BLEU. With fewer n-grams to compare, and SwinBERT not being able to output exact annotated words, scores are decreased. We also observed the capability of SwinBERT to understand punctuation in sentences, as we get slightly better scores if we do not remove the punctuation signs. In conclusion, the best combination in terms of BLEU-1 is the one where we maintained larger descriptions with a unified lexicon.

It can be observed that the results for BLEU-1 are not as high as expected, given the performance of SwinBERT on other mainstream datasets. This can be explained by two factors: the goal of the Charades dataset, which is designed for visual analysis, and the nature of BLEU as a metric that was originally conceived for translation tasks and does not take into account the semantic similarity of words but only exact matches.

Table 5.2 shows the previously described results in terms of precision and recall obtained from BERT Score. Here, we can see slightly higher results compared to Table 5.1 due to BERT’s ability to capture semantic similarity. In this table, we can observe more homogeneous values between different pretrainings because sentence length is not as important as it was for the BLEU calculation.

5.2. Object Recognition evaluation

This section will focus on evaluating the Object Recognition architecture used to extract objects from videos. As mentioned earlier, due to time constraints, our evaluation will be limited to 56 specific object classes. Therefore, all stages of the pipeline and experiments will be designed to test the recognition capability of these mentioned classes.

5.2.1. Description

In this third proposal, our goal is to evaluate the Object Recognition pipeline using a combination of two fundamental models: a Video Captioning model and a pure Object Recognition model. The Video Captioning model will capture the most relevant items and objects from the video, while the Object Recognition model will detect not only the main items but also the surrounding objects in the scene. These models will complement each other to enhance video comprehension and increase the number of recognized objects. The pipeline for this evaluation consists of three stages: Model execution, object extraction, and evaluation. To conduct the experimentation, we will use a subset of 1,430 videos from the Charades trainset. These videos have an annotated set of objects, and at least one of the objects must belong to our COCO2017 classes subset. For this purpose we will randomly select 3,000 videos from the Charades trainset and then we will filter the videos from this subset to only include those with at least one object from our COCO2017 classes subset. Additionally, we will filter the annotations themselves to only include objects belonging to our COCO2017 classes subset.

In the first stage of the pipeline, we will provide the input video to both the Video Captioning and Object Recognition models to compute their outputs. For the Video Captioning task, we will use the SwinBERT model discussed earlier, and for the Object Recognition task, we will use YOLOv7. We will conduct different experiments using the VATEX, MSRVTT, and MSVD SwinBERT pretrainings, while keeping the YOLOv7 model pretraining constant to COCO2017. A confidence threshold of 0.7 will be used for YOLOv7 predictions, as explained in Subsection 4.2.4.

In the second stage of the pipeline, we will extract objects from both model predictions and create a unified set of objects. Since SwinBERT outputs natural language descriptions, we need to process these texts to extract objects. For this purpose, we will use the objects extractor module described in Subsection 4.2.3. This module makes use of the pretrained model *word2vec-google-news-300* available from the Gensim API to obtain vector embeddings for each word in the generated description. We will compare these embeddings with the embeddings of objects belonging to the subset of 56 classes from COCO2017. This comparison will allow us to identify matches between the description's words and the COCO2017 classes, representing these matches the objects present in the description. We will combine the objects extracted from SwinBERT descriptions with the objects detected by YOLOv7, removing any duplicates.

Finally, in the third stage of the pipeline, we will evaluate the results using the Recall score. We will compare the set of output objects from the second stage with the annotated objects in the Charades dataset. Similar to the second stage, we will use the *word2vec-google-news-300* pretrained model to compute embeddings and perform object matching between the annotated and detected sets of objects. In this stage, we will use a similarity threshold of 0.62 to consider a pair of embeddings as a match. Selection of this threshold was explained in Subsection 4.2.3. The complete pipeline for this experimentation is shown in Figure 5.6.

5.2.2. Results

In Table 5.3, the Recall results for the conducted experimentation of the Object Recognition pipeline on the subset of 1,430 filtered videos from Charades are presented. The highest score in the table is highlighted in bold. Additionally, Figure 5.7 showcases the objects recognized

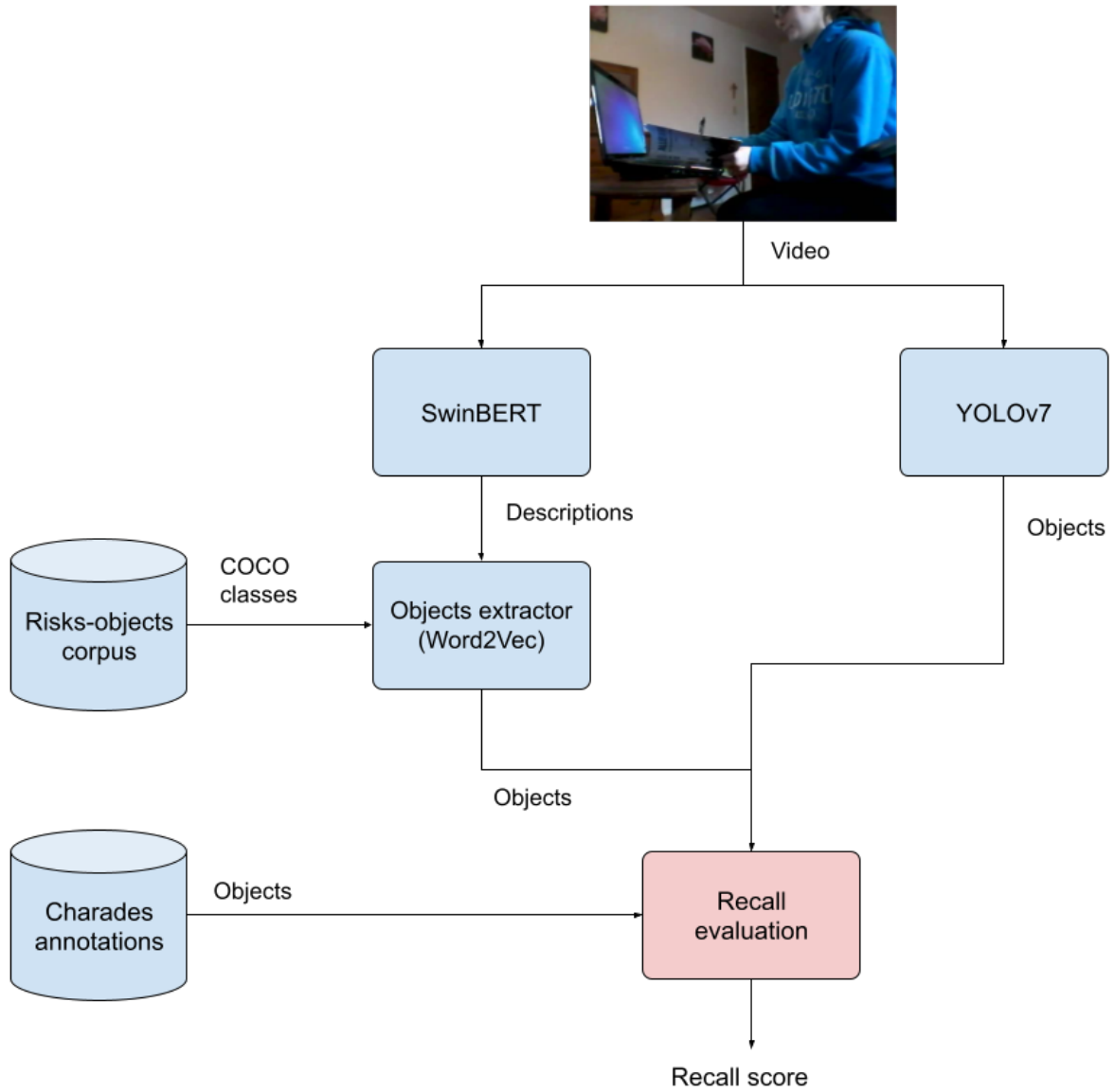


Figure 5.6: Pipeline used for computing the evaluation of the Object Recognition proposed architecture. COCO classes are extracted from the Risks-objects corpus as this corpus is based on our subset of COCO’s 56 classes.



YOLO: Remote, cellphone, cup, chair, tv, laptop
MSVD: -
MSR-VTT: Couch
VATEX: Couch, book

Figure 5.7: Objects extracted from YOLO and different SwinBERT pretrainings over an input Charades video. Observe confusion recognising the screen as a laptop.

by YOLOv7 and the different SwinBERT pretrainings used over a sample image.

In the previously mentioned table, we can observe the differences in results when using the three different SwinBERT pretrainings discussed throughout the thesis. We can also see the impact of incorporating the complementary YOLOv7 model within the architecture. The results under the column "Not YOLO" represent the Recall score when the final set of detected objects consists only of the objects extracted from SwinBERT descriptions.

As we can observe, differences between the results obtained from the three SwinBERT pretrainings, when YOLOv7 model is not used, are consistent with the findings in Subsection 5.1.2, where we evaluated the generated descriptions using BLEU-1 and BERT Score. VATEX pretraining achieves the highest score due to its denser descriptions, which allow for the extraction of more objects as more tokens are involved in the output. Furthermore, the table reflects the significantly positive impact of complementing SwinBERT-extracted objects with the purely Object Recognition model YOLOv7. We can observe how the addition of YOLOv7 detections increases scores up to five times, particularly when using the MSVD SwinBERT pretraining. Moreover, the last row of the table represents the evaluation when using only detections obtained from YOLOv7. By examining the results when YOLOv7 is used, we can conclude that the majority of the relevant detections are computed by this model, while SwinBERT acts as a secondary model that provides a smaller contribution to the final score. Consistent with previous experiments, we have demonstrated that the most effective SwinBERT pretraining is VATEX, yielding the highest Recall score for Object Recognition when combined with YOLOv7.

Although various combinations have been tested, the maximum Recall score achieved is **0.5719**. This result can be attributed to several factors that imposed limitations on this experimentation. The first factor is the quality of the videos in the Charades dataset. As mentioned earlier, Charades was compiled using crowdsourcing, resulting in videos of varying quality. This poor quality in some of the videos within our subset makes it challenging to accurately recognize objects in some scenes. The second factor is related to the limited annotations provided by Charades regarding objects. Although Charades offers annotations for the most relevant objects in the video, it does not cover the entire set of objects present in the scenes. Consequently, a significant number of objects detected by the Object Recognition architecture do not contribute to the computation of the final score because these objects are not included in the annotation set. In Figures C.2 and C.1 of the Appendix III are shown demonstrations of these factors that are limiting the scores retrieved in the experimentation.

Pretraining	YOLO	Not YOLO
MSVD	0.5351	0.1046
MSR-VTT	0.5359	0.0915
VATEX	0.5719	0.2047
-	0.5082	-

Table 5.3: Recall evaluation of the Object Recognition pipeline over 1430 subset of videos from Charades dataset. Last row represents Recall when only using YOLO detections.

5.3. Risks Assessment

In this final experiment, we will pursue the main objective proposed for this thesis, which is the development of a pipeline for identifying potential risks associated with the environment depicted in the input video. This pipeline serves as a practical application for all the research and experimentation conducted throughout the thesis.

5.3.1. Description

The objective of the proposed pipeline in this subsection is to assess potential risks in input videos based on the objects present in the depicted environment. This pipeline will analyze the input video and generate a list of risks associated with the detected objects. Each risk will be assigned a level of relevance based on the importance of the associated object within the scene. We will utilize both the ETRI-Activity3D and Charades datasets to obtain execution samples. We will use Charades as this is the main dataset used during the complete experimentation, while we will also provide an additional execution sample from ETRI-Activity3D given its nearest application to create systems involving the dementia motivation introduced at the beginning of the thesis.

Similar to Section 5.2, where the Object Recognition pipeline was presented, this pipeline will consist of three stages: Model execution, object extraction, and Risk Assessment. In the

first stage, we will obtain the outputs from the SwinBERT and YOLOv7 models. YOLOv7 will utilize the COCO2017 pretraining with a confidence threshold of 0.7, and SwinBERT will employ the VATEX pretraining, as this combination has shown to yield superior results. In the second stage, the object extraction module, with the Gensim pretrained model *word2vec-google-news-300* as its backbone, will be employed to extract objects belonging to our COCO2017 classes subset from the SwinBERT descriptions. Finally, in the third stage, the Risk Assessment will be performed using the Risks Matching module described in Subsection 4.2.5. In this module, objects provided by YOLOv7 and SwinBERT descriptions will be assigned a level of relevance, and risks will be generated based on the complete set of detected objects. The level of relevance assigned to an object will be associated with the risks involved. The complete proposed pipeline is depicted in Figure 5.8.

5.3.2. Results

Now, the proposed Risk Assessment pipeline will be tested on one sample from the Charades and ETRI-Activity3D datasets. For each sample, the detected risks will be shown, along with the relevance level of the risk and the object from which the risk was extracted. Additionally, a figure will be included in which the detected objects are highlighted. Only one sample from each dataset is provided because the purpose of this experimentation is to provide an intuitive understanding of how the Risks Assessment pipeline functions, rather than conducting a formal evaluation of it. The risks obtained are entirely based on the recognized objects, so evaluating this pipeline would entail assessing the Object Recognition pipeline, which has already been evaluated in previous sections.

Figure 5.9 presents the risks detected in a Charades input video, while Figure 5.10 shows the detections obtained from the previously analyzed video. Figure 5.12 displays the risks detected in an ETRI-Activity3D input video, and Figure 5.11 illustrates the detections associated with this video.

In Section 5.2, we mentioned that the quality of videos was an important factor when drawing conclusions from the results. Since this factor poses challenges in the Object Recognition task, it consequently affects the Risks Assessment since it is based on Object Recognition. This, combined with the computer vision's ongoing challenge of differentiating objects that share a high number of common features, results in the architecture occasionally confusing similar objects. This can be observed in Figure 5.9, where the pipeline confuses the screen with a laptop. It should be noted that despite this confusion, the obtained risks are still valid, as objects with similar characteristics are expected to yield similar risks.

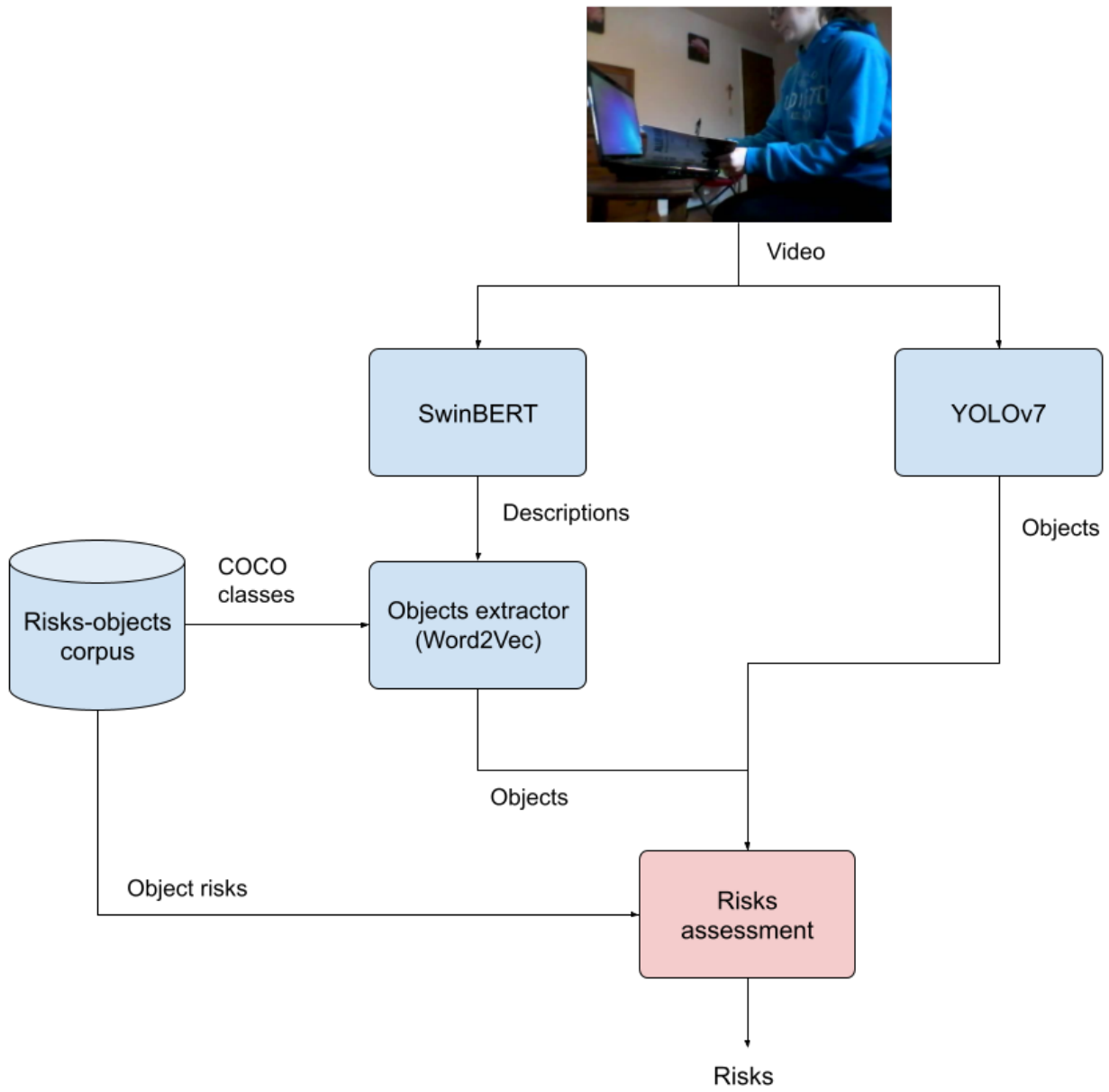


Figure 5.8: Pipeline used to compute Risks Assessment.



Risk 1: If the person makes an imprecise movement, he/she may fall out of the chair. Associated with “Couch” [Lv. 2] and “Chair” [Lv.1]

Risk 2: If the object is positioned in an unexpected location within the room, it can potentially cause individuals to trip over it, leading to accidental falls. Associated with “Book” [Lv.2] and “Chair” [Lv.1]

Risk 3: If the object collides with something or is not used properly, its content can be spilt. Associated with “Cup” [Lv.1]

Risk 4: If the object collides with something, it can be converted into dangerous pieces which can potentially injure the person. Associated with “Cup” [Lv.1], “TV” [Lv.1] and “Laptop” [Lv.1]

Risk 5: If the object remains powered on for a long time, it can lead to high energy consumption expenses.. Associated with “TV” [Lv.1]

Risk 6: If a short circuit occurs, it can produce an electric shock that has the potential to injure the person. Associated with “Remote” [Lv.1], “Cell phone” [Lv.1], “TV” [Lv.1] and “Laptop” [Lv.1]

Figure 5.9: Risks Assessment over a Charades dataset input video. Although the pipeline momentarily confuses the TV (screen computer) with a laptop all the risks detected are still valid.

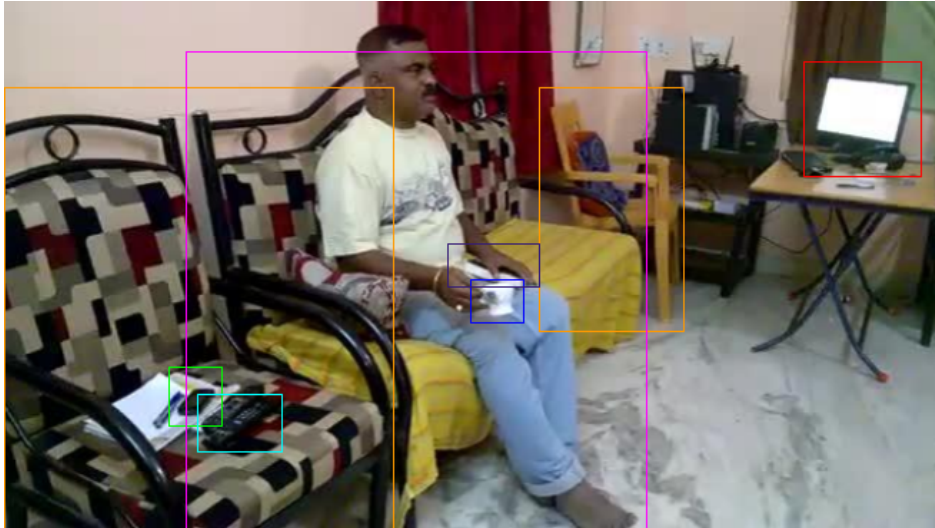


Figure 5.10: Detections associated to the Figure 5.9. In red the TV/Laptop, in orange the chairs, in green the cellphone, in cyan blue the remote, in dark blue the cup, in purple the book and in magenta the couch.



Figure 5.11: Detections associated to the Figure 5.12. In cyan blue the bottle, in green the refrigerator



Risk 1: If the object's door remains opened for a long time it can provoke high expenses in energy consumption. Associated with "Refrigerator" [Lv. 3]

Risk 2: If the object's door remains opened for a long time it can provoke food to spoil. Associated with "Refrigerator" [Lv. 3]

Risk 3: If the object collides with something or is not used properly, its content can be spilt. Associated with "Bottle" [Lv. 1]

Risk 4: If the object is made of a fragile material and it collides with something, it can be converted into dangerous pieces which can potentially injure the person. Associated with "Bottle" [Lv. 1]

Figure 5.12: Risks Assessment over an ETRI-Activity3D input video.

6. Conclusions

In this last chapter we will provide a synopsis of all the contributions presented in this thesis and we will review future works that can be carried out in following studies. In section 6.1 we will summarize highlighted points of thesis, then in section 6.2 we will revise some topics that can be further studied in future works and finally in section 6.3 we will present publications done in consequence to the development of the thesis.

6.1. Conclusions

In this thesis, we conducted a comprehensive review of the state-of-the-art in the Video Captioning task, which involves generating a textual description from an input video. Based on this review, we proposed a Risks Assessment pipeline that combines Video Captioning with Object Recognition. This pipeline has potential applications in various fields, such as the development of personalized assistants to support individuals with special needs in their daily lives. Additionally, we introduced three evaluation architectures to assess the different models within the Risks Assessment pipeline. These architectures utilized diverse components, including different datasets and metrics during the evaluation stage. Notably, SwinBERT was consistently used as the Video Captioning model across all architectures due to its demonstrated performance and effectiveness in general contexts.

In the first evaluation architecture, we combined SwinBERT as the captioning model with Zero-shot classification for evaluation purposes. Through this experiment, we demonstrated the challenges in classifying generated descriptions from the ETRI-Activity3D dataset into a single label due to the large set of different labels provided by the dataset. Consequently, we concluded that this architecture is not suitable for our intended purposes without further studies. Evaluation results for this architecture were given in terms of Recall score.

In the second evaluation architecture, we proposed utilizing SwinBERT with different pre-trainings (MSVD, VATEX, and MSR-VTT) and applied various lexical transformations to the outputs generated from processing videos from the Charades dataset, which contains annotated descriptions unlike ETRI-Activity3D. We observed differences among these pre-trainings, with VATEX showing the best performance due to its ability to produce more extensive and detailed descriptions. We found that lexical transformations that unified the sentence structure without removing semantic information (such as lemmatization or lemmatization with punctuation removal) achieved the highest scores. Additionally, we highlighted the challenges posed by the Charades dataset for computer vision description generation and evaluated a sample of 3,000 videos using BLEU-1 and BERT Score metrics. BERT Score was deemed more suitable for evaluation as it captures semantic similarity over lexical information. We concluded that this architecture is the most suitable among the evaluated ones for fulfilling the description generation task.

In the third evaluation architecture, we proposed an Object Recognition pipeline composed

of SwinBERT and complemented by YOLOv7, which enables a higher number of detections compared to SwinBERT alone, which focuses on the most relevant objects. We utilized a subset of 1,430 videos from the Charades trainset, filtered based on the existence of at least one object belonging to our COCO2017 classes subset. In this evaluation, we emphasized the importance and relevance of YOLOv7 for object extraction and concluded that the best configuration for this architecture was to use VATEX pretraining for SwinBERT complemented by the YOLOv7 model. VATEX pretraining yielded superior results as it produced denser descriptions, which facilitated the inclusion of more objects within descriptions. We also discussed two factors that limited the results: the poor quality of some Charades videos and the lack of appropriate object annotations. Evaluation results were presented in terms of Recall score.

Finally, we introduced the proposed Risks Assessment pipeline, which outputs risks along with their relevance levels and the associated objects based on an input video. For this pipeline, we utilized an architecture consisting of SwinBERT pretrained on VATEX and YOLOv7 pretrained on COCO2017. We provided execution samples using input videos from the Charades and ETRI-Activity3D datasets, and we explained that the typical problem in computer vision of confusing very similar objects would not significantly impact our assessment, as similar objects tend to involve similar risks.

6.2. Future works

In this section, we will provide an overview of works, studies, and enhancements that will be considered for future development. These potential works are relevant for increasing the performance of the different stages involved in the proposed pipelines, ultimately leading to better results in the Risks Assessment task. The improvements resulting from these works can potentially enable the utilization of the proposed pipeline in production environments, such as personalized assistant systems for individuals with special needs.

- **Dense Video Captioning:** This task aims to obtain more detailed descriptions from videos compared to Video Captioning. Future studies will focus on exploring this task and researching models that can generate denser descriptions. This will increase the possibilities of extracting a higher number of objects from the descriptions.
 - **Zero-shot classification:** In the second proposed evaluation architecture, it was concluded that using Zero-shot classification as an evaluation method was not suitable for assessing descriptions. Further studies are needed to assess its suitability, including the exploration of different models for the labeling process. Additionally, the inclusion of a training stage for the labeling model can be considered.
 - **Subset of object classes:** The Object Recognition task was limited to recognizing up to 56 different object classes due to the YOLOv7 pretraining on COCO2017. For future work, a training stage for YOLOv7 on objects commonly found in home environments should be considered to increase the number of detectable classes.
 - **Alternative object segmentation methods:** “*Segment Everything Everywhere All at Once*” (Zou et al., 2023) is a recently published paper that demonstrates a system
-

capable of segmenting visual content based on prompts. Future studies could explore the use of SwinBERT descriptions as reference prompts to guide the segmentation performed by such a system.

- **SwinBERT contextual information:** SwinBERT descriptions provide information about the context, events, and actions occurring in the scene. Future studies can leverage this information to complement the obtained detections and generate a list of risks that not only depend on the detected objects but also consider the context of the scene.
- **Object states:** The dangerousness of an object can vary depending on its state. Further studies on detecting object states are necessary to include this information in the Risks Assessment, enabling the exclusion of risks associated with states of objects not present in the input scene.

6.3. Publications

As a consequence of the development of this thesis we were able to compose and release a publication which covers some part of the experimentation presented in this study. The work was released in the 18th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2023) entitled as “*Indoor Scenes Video Captioning*”.

Bibliography

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... Zheng, X. (2016). *Tensorflow: A system for large-scale machine learning*. Retrieved from <https://arxiv.org/abs/1605.08695>
- Afif, M., Ayachi, R., Said, Y., & Atri, M. (2020, Jun 01). Deep learning based application for indoor scene recognition. *Neural Processing Letters*, *51*(3), 2827-2837. Retrieved from <https://doi.org/10.1007/s11063-020-10231-w> doi: 10.1007/s11063-020-10231-w
- Alzheimer's disease international, A. (2011). *The benefits of early diagnosis and intervention*. Retrieved from <https://www.alzint.org/u/WorldAlzheimerReport2011.pdf>
- Arriaga, O., Plöger, P., & Valdenegro-Toro, M. (2017). *Image captioning and classification of dangerous situations*. arXiv. Retrieved from <https://arxiv.org/abs/1711.02578> doi: 10.48550/ARXIV.1711.02578
- Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural machine translation by jointly learning to align and translate*. arXiv. Retrieved from <https://arxiv.org/abs/1409.0473> doi: 10.48550/ARXIV.1409.0473
- Becker, J. T., Boller, F., Lopez, O. L., Saxton, J., & McGonigle, K. L. (1994, June). The natural history of alzheimer's disease. description of study cohort and accuracy of diagnosis. *Arch Neurol*, *51*(6), 585–594.
- Burrell, P. R., & Folarin, B. O. (1997, Dec 01). The impact of neural networks in finance. *Neural Computing & Applications*, *6*(4), 193-200. Retrieved from <https://doi.org/10.1007/BF01501506> doi: 10.1007/BF01501506
- Carreira, J., Noland, E., Banki-Horvath, A., Hillier, C., & Zisserman, A. (2018). *A short note about kinetics-600*. Retrieved from <https://arxiv.org/abs/1808.01340>
- Chen, D., & Dolan, W. (2011, June). Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies* (pp. 190–200). Portland, Oregon, USA: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/P11-1020>
- Chen, F., Ji, R., Dai, C., Sun, X., Lin, C.-W., Ji, J., ... Cao, L. (2019). *Semantic-aware image deblurring*. arXiv. Retrieved from <https://arxiv.org/abs/1910.03853> doi: 10.48550/ARXIV.1910.03853
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning phrase representations using rnn encoder-decoder for statistical machine translation*. arXiv. Retrieved from <https://arxiv.org/abs/1406.1078> doi: 10.48550/ARXIV.1406.1078

- Chollet, F. (2016). *Xception: Deep learning with depthwise separable convolutions*. arXiv. Retrieved from <https://arxiv.org/abs/1610.02357> doi: 10.48550/ARXIV.1610.02357
- Culjak, I., Abram, D., Pribanic, T., Dzapo, H., & Cifrek, M. (2012). A brief introduction to opencv. In *2012 proceedings of the 35th international convention mipro* (p. 1725-1730).
- Deng, C., Chen, S., Chen, D., He, Y., & Wu, Q. (2021). Sketch, ground, and refine: Top-down dense video captioning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 234-243). doi: 10.1109/CVPR46437.2021.00030
- Deorukhkar, K., & Ket, S. (2022, Jan 01). A detailed review of prevailing image captioning methods using deep learning techniques. *Multimedia Tools and Applications*, 81(1), 1313-1336. Retrieved from <https://doi.org/10.1007/s11042-021-11293-1> doi: 10.1007/s11042-021-11293-1
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *Bert: Pre-training of deep bidirectional transformers for language understanding*. Retrieved from <https://arxiv.org/abs/1810.04805>
- Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., & Sun, J. (2021). *Repvgg: Making vgg-style convnets great again*. Retrieved from <https://arxiv.org/abs/2101.03697>
- Fernández Montenegro, J. M., Villarini, B., Angelopoulou, A., Kapetanios, E., Garcia-Rodriguez, J., & Argyriou, V. (2020). A survey of alzheimer's disease early diagnosis methods for cognitive assessment. *Sensors*, 20(24). Retrieved from <https://www.mdpi.com/1424-8220/20/24/7292> doi: 10.3390/s20247292
- Fudholi, D. H., & Nayoan, R. A. (2022, Aug 06). The role of transformer-based image captioning for indoor environment visual understanding. *International Journal of Computing and Digital Systems*, 12(1), 479-488. doi: <https://dx.doi.org/10.12785/ijcds/120138>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition*. arXiv. Retrieved from <https://arxiv.org/abs/1512.03385> doi: 10.48550/ARXIV.1512.03385
- Herd, P., Carr, D., & Roan, C. (2014, February). Cohort profile: Wisconsin longitudinal study (WLS). *Int J Epidemiol*, 43(1), 34-41.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780. doi: 10.1162/neco.1997.9.8.1735
- Huang, S., Usvyatsov, M., & Schindler, K. (2020). Indoor scene recognition in 3d. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (p. 8041-8048). doi: 10.1109/IROS45743.2020.9341580
- Islam, S., Dash, A., Seum, A., Raj, A., Hossain, T., & Shah, F. (2021, 04). Exploring video captioning techniques: A comprehensive survey on deep learning methods. *SN Computer Science*, 2. doi: 10.1007/s42979-021-00487-x
-

- Jain, V., Al-Turjman, F., Chaudhary, G., Nayar, D., Gupta, V., & Kumar, A. (2022, Oct 01). Video captioning: a review of theory, techniques and practices. *Multimedia Tools and Applications*, 81(25), 35619-35653. Retrieved from <https://doi.org/10.1007/s11042-021-11878-w> doi: 10.1007/s11042-021-11878-w
- Jang, J., Kim, D., Park, C., Jang, M., Lee, J., & Kim, J. (2020). *Etri-activity3d: A large-scale rgb-d dataset for robots to recognize daily activities of the elderly*. Retrieved from <https://arxiv.org/abs/2003.01920>
- Jin, T., Li, Y., & Zhang, Z. (2019). Recurrent convolutional video captioning with global and local attention. *Neurocomputing*, 370, 118-127. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0925231219311804> doi: <https://doi.org/10.1016/j.neucom.2019.08.042>
- Karakostas, A., Briassouli, A., Avgerinakis, K., Kompatsiaris, I., & Tsolaki, M. (2016). *The dem@care experiments and datasets: a technical report*. Retrieved from <https://arxiv.org/abs/1701.01142>
- Kempler, D., Curtiss, S., & Jackson, C. (1987, September). Syntactic preservation in alzheimer's disease. *J Speech Hear Res*, 30(3), 343-350.
- Khurana, D., Koli, A., Khatter, K., & Singh, S. (2023, Jan 01). Natural language processing: state of the art, current trends and challenges. *Multimedia Tools and Applications*, 82(3), 3713-3744. Retrieved from <https://doi.org/10.1007/s11042-022-13428-4> doi: 10.1007/s11042-022-13428-4
- Koenig, L. N., Day, G. S., Salter, A., Keefe, S., Marple, L. M., Long, J., ... Benzinger, T. L. (2020). Select atrophied regions in alzheimer disease (sara): An improved volumetric model for identifying alzheimer disease dementia. *NeuroImage: Clinical*, 26, 102248. Retrieved from <https://www.sciencedirect.com/science/article/pii/S2213158220300851> doi: <https://doi.org/10.1016/j.nicl.2020.102248>
- Kojima, A., Tamura, T., & Fukunaga, K. (2002, Nov 01). Natural language description of human activities from video images based on concept hierarchy of actions. *International Journal of Computer Vision*, 50(2), 171-184. Retrieved from <https://doi.org/10.1023/A:1020346032608> doi: 10.1023/A:1020346032608
- Koller, D., Heinze, N., & Nagel, H. (1991). Algorithmic characterization of vehicle trajectories from image sequences by motion verbs. In *Proceedings. 1991 ieee computer society conference on computer vision and pattern recognition* (p. 90-95). doi: 10.1109/CVPR.1991.139667
- Krishna, R., Hata, K., Ren, F., Fei-Fei, L., & Niebles, J. C. (2017). *Dense-captioning events in videos*. arXiv. Retrieved from <https://arxiv.org/abs/1705.00754> doi: 10.48550/ARXIV.1705.00754
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017, may). Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6), 84-90. Retrieved from <https://doi.org/10.1145/3065386> doi: 10.1145/3065386
-

- Krogh, A. (2008, Feb 01). What are artificial neural networks? *Nature Biotechnology*, 26(2), 195-197. Retrieved from <https://doi.org/10.1038/nbt1386> doi: 10.1038/nbt1386
- Kumar, P., Lai, S. H., Wong, J. K., Mohd, N. S., Kamal, M. R., Afan, H. A., ... El-Shafie, A. (2020). Review of nitrogen compounds prediction in water bodies using artificial neural networks and other models. *Sustainability*, 12(11). Retrieved from <https://www.mdpi.com/2071-1050/12/11/4359> doi: 10.3390/su12114359
- LaMontagne, P. J., Benzinger, T. L., Morris, J. C., Keefe, S., Hornbeck, R., Xiong, C., ... Marcus, D. (2019). Oasis-3: Longitudinal neuroimaging, clinical, and cognitive dataset for normal aging and alzheimer disease. *medRxiv*. Retrieved from <https://www.medrxiv.org/content/early/2019/12/15/2019.12.13.19014902> doi: 10.1101/2019.12.13.19014902
- Li, X., Zhou, Z., Chen, L., & Gao, L. (2019, Mar 01). Residual attention-based lstm for video captioning. *World Wide Web*, 22(2), 621-636. Retrieved from <https://doi.org/10.1007/s11280-018-0531-z> doi: 10.1007/s11280-018-0531-z
- Lin, D., Kong, C., Fidler, S., & Urtasun, R. (2015). *Generating multi-sentence lingual descriptions of indoor scenes*. Retrieved from <https://arxiv.org/abs/1503.00064>
- Lin, K., Li, L., Lin, C.-C., Ahmed, F., Gan, Z., Liu, Z., ... Wang, L. (2022). *Swinbert: End-to-end transformers with sparse attention for video captioning*. Retrieved from <https://arxiv.org/abs/2111.13196>
- Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., ... Dollár, P. (2015). *Microsoft coco: Common objects in context*. Retrieved from <https://arxiv.org/abs/1405.0312>
- Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., & Hu, H. (2021). *Video swin transformer*. Retrieved from <https://arxiv.org/abs/2106.13230>
- Manay, S. P., Yaligar, S. A., Thathva Sri Sai Reddy, Y., & Saunshimath, N. J. (2022). Image captioning for the visually impaired. In N. R. Shetty, L. M. Patnaik, H. C. Nagaraj, P. N. Hamsavath, & N. Nalini (Eds.), *Emerging research in computing, information, communication and applications* (pp. 511-522). Singapore: Springer Singapore.
- Marcus, D. S., Fotenos, A. F., Csernansky, J. G., Morris, J. C., & Buckner, R. L. (2010, 12). Open Access Series of Imaging Studies: Longitudinal MRI Data in Nondemented and Demented Older Adults. *Journal of Cognitive Neuroscience*, 22(12), 2677-2684. Retrieved from <https://doi.org/10.1162/jocn.2009.21407> doi: 10.1162/jocn.2009.21407
- Marcus, D. S., Wang, T. H., Parker, J., Csernansky, J. G., Morris, J. C., & Buckner, R. L. (2007, 09). Open Access Series of Imaging Studies (OASIS): Cross-sectional MRI Data in Young, Middle Aged, Nondemented, and Demented Older Adults. *Journal of Cognitive Neuroscience*, 19(9), 1498-1507. Retrieved from <https://doi.org/10.1162/jocn.2007.19.9.1498> doi: 10.1162/jocn.2007.19.9.1498
- Mishev, K., Gjorgjevikj, A., Vodenska, I., Chitkushev, L., & Trajanov, D. (2020, 07). Evaluation of sentiment analysis in finance: From lexicons to transformers. *IEEE Access*, PP, 1-1. doi: 10.1109/ACCESS.2020.3009626
-

- Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002, July). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the association for computational linguistics* (pp. 311–318). Philadelphia, Pennsylvania, USA: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/P02-1040> doi: 10.3115/1073083.1073135
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). *Pytorch: An imperative style, high-performance deep learning library*. Retrieved from <https://arxiv.org/abs/1912.01703>
- Pei, W., Zhang, J., Wang, X., Ke, L., Shen, X., & Tai, Y.-W. (2019). *Memory-attended recurrent network for video captioning*. arXiv. Retrieved from <https://arxiv.org/abs/1905.03966> doi: 10.48550/ARXIV.1905.03966
- Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., & Torralba, A. (2018). *Virtualhome: Simulating household activities via programs*. Retrieved from <https://arxiv.org/abs/1806.07011>
- Puthusseryppady, V., Morrissey, S., Spiers, H., Patel, M., & Hornberger, M. (2022, Aug 04). Predicting real world spatial disorientation in alzheimer’s disease patients using virtual reality navigation tests. *Scientific Reports*, 12(1), 13397. Retrieved from <https://doi.org/10.1038/s41598-022-17634-w> doi: 10.1038/s41598-022-17634-w
- Řehůřek, R., & Sojka, P. (2010, May 22). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA. Retrieved from <http://is.muni.cz/publication/884893/en>
- Sigurdsson, G. A., Varol, G., Wang, X., Farhadi, A., Laptev, I., & Gupta, A. (2016). *Hollywood in homes: Crowdsourcing data collection for activity understanding*. Retrieved from <https://arxiv.org/abs/1604.01753>
- Simonyan, K., & Zisserman, A. (2014). *Very deep convolutional networks for large-scale image recognition*. arXiv. Retrieved from <https://arxiv.org/abs/1409.1556> doi: 10.48550/ARXIV.1409.1556
- Sordo, M. (2002). Introduction to neural networks in healthcare. *Open clinical: Knowledge management for medical care*.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2014). *Going deeper with convolutions*. arXiv. Retrieved from <https://arxiv.org/abs/1409.4842> doi: 10.48550/ARXIV.1409.4842
- Timilsina, S., Sharma, S., & Aryal, J. (2019, 12). Mapping urban trees within cadastral parcels using an object-based convolutional neural network. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, IV-5/W2*, 111-117. doi: 10.5194/isprs-annals-IV-5-W2-111-2019
-

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). *Attention is all you need*. arXiv. Retrieved from <https://arxiv.org/abs/1706.03762> doi: 10.48550/ARXIV.1706.03762
- Villamizar Torres, J., & Lizarazo, P. (2019). *Reconocimiento de actividades humanas usando redes de gran memoria de corto plazo* (Doctoral dissertation). doi: 10.13140/RG.2.2.22331.85282
- Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). *Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. Retrieved from <https://arxiv.org/abs/2207.02696>
- Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2021). *You only learn one representation: Unified network for multiple tasks*. Retrieved from <https://arxiv.org/abs/2105.04206>
- Wang, X., Wu, J., Chen, J., Li, L., Wang, Y.-F., & Wang, W. Y. (2020). *Vatex: A large-scale, high-quality multilingual dataset for video-and-language research*. Retrieved from <https://arxiv.org/abs/1904.03493>
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2020). *Huggingface's transformers: State-of-the-art natural language processing*. Retrieved from <https://arxiv.org/abs/1910.03771>
- World Health Organization, W. (2021). *Global status report on the public health response to dementia*. Retrieved from <https://www.who.int/publications/i/item/9789240033245>
- Xu, J., Mei, T., Yao, T., & Rui, Y. (2016). Msr-vtt: A large video description dataset for bridging video and language. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 5288-5296). doi: 10.1109/CVPR.2016.571
- Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). *Bertscore: Evaluating text generation with bert*. Retrieved from <https://arxiv.org/abs/1904.09675>
- Zhou, L., Zhou, Y., Corso, J. J., Socher, R., & Xiong, C. (2018). *End-to-end dense video captioning with masked transformer*. arXiv. Retrieved from <https://arxiv.org/abs/1804.00819> doi: 10.48550/ARXIV.1804.00819
- Zou, X., Yang, J., Zhang, H., Li, F., Li, L., Gao, J., & Lee, Y. J. (2023). *Segment everything everywhere all at once*. Retrieved from <https://arxiv.org/abs/2304.06718v1>
-

List of Acronyms and Abbreviations

AAL	Ambient Assisted Living.
BERT	Bidirectional Encoder Representations from Transformers.
CNN	Convolutional Neural Network.
DL	Deep Learning.
DLSI	Departamento de Lenguajes y Sistemas Informáticos.
DTIC	Departamento de Tecnología Informática y Computación.
GRU	Gate Recurrent Unit.
IoU	Intersection over Union.
LSTM	Long Short-Term Memory.
mAP	Mean Average Precision.
MCI	Mild Cognitive Impairment.
ML	Machine Learning.
MoDeAss	Monitoring and Detection of human behaviors for personalized assistance and early disease detection, A2HUMPA in Spanish.
MRI	Magnetic Resonance Imaging.
NAS	Network Attached Storage.
NLP	Natural Language Processing.
R-CNN	Regions with Convolutional Neural Network features.
RNN	Recurrent Neural Network.
SOCO 2023	18th International Conference on Soft Computing Models in Industrial and Environmental Applications.
SSH	Secure Shell.
SVO	Subject, Verb and Object.

A. Appendix I

Action	Score	Action	Score
eating food with a fork	31.25%	pouring water into a cup	0.0%
taking medicine	0.0%	drinking water	12.5%
putting food in the fridge/taking food from the fridge	0.0%	trimming vegetables	0.0%
peeling fruit	0.0%	using a gas stove	0.0%
cutting vegetable on the cutting board	0.0%	brushing teeth	0.0%
washing hands	18.75%	washing face	0.0%
wiping face with a towel	0.0%	putting on cosmetics	0.0%
putting on lipstick	0.0%	brushing hair	50.0%
blow drying hair	0.0%	putting on a jacket	0.0%
taking off a jacket	0.0%	putting on/taking off shoes	0.0%
putting on/taking off glasses	0.0%	washing the dishes	0.0%
vacuuming the floor	0.0%	scrubbing the floor with a rag	81.25%
wiping off the dining table	12.5%	rubbing up furniture	25.0%
spreading bedding/folding bedding	100.0%	washing a towel by hands	0.0%
hanging out laundry	0.0%	looking around for something	4.17%
using a remote control	0.0%	reading a book	0.0%
reading a newspaper	75.0%	handwriting	6.25%

Table A.1: Recall percentage obtained from testing Zero-shot classification for each of the ETRI-Activity3D dataset’s actions (From action 1-34).

Action	Score	Action	Score
talking on the phone	50.0%	playing with a mobile phone	0.0%
using a computer	0.0%	smoking	0.0%
clapping	0.0%	rubbing face with hands	0.0%
doing freehand exercise	6.25%	doing neck roll exercise	0.0%
massaging a shoulder oneself	0.0%	taking a bow	0.0%
talking to each other	12.5%	handshaking	12.5%
hugging each other	0.0%	fighting each other	0.0%
waving a hand	0.0%	flapping a hand up and down (beckoning)	0.0%
pointing with a finger	0.0%	opening the door and walking in	12.5%
fallen on the floor	29.17%	sitting up/standing up	33.33%
lying down	45.83%	-	-

Table A.2: Recall percentage obtained from testing Zero-shot classification for each of the ETRI-Activity3D dataset’s actions (From action 35-55).

B. Appendix II

B.1. Objects corpus

```
1  [  
2    {  
3      "object": "ball",  
4      "properties": ["playable", "cause_stumbling"],  
5      "specific_risks": []  
6    },  
7    {  
8      "object": "bicycle",  
9      "properties": ["cause_stumbling"],  
10     "specific_risks": []  
11   },  
12   {  
13     "object": "bench",  
14     "properties": ["cause_stumbling"],  
15     "specific_risks": []  
16   },  
17   {  
18     "object": "bird",  
19     "properties": ["alive", "break_objects"],  
20     "specific_risks": []  
21   },  
22   {  
23     "object": "cat",  
24     "properties": ["alive", "break_objects"],  
25     "specific_risks": []  
26   },  
27   {  
28     "object": "dog",  
29     "properties": ["alive", "break_objects"],  
30     "specific_risks": []  
31   },  
32   {  
33     "object": "umbrella",  
34     "properties": ["cause_stumbling"],  
35     "specific_risks": []  
36   },  
37   ...
```

Figure B.1: Fragment of the objects file which comprises the Risks-objects corpus (Object 1-7).

```
1  ...
2  {
3    "object": "handbag",
4    "properties": ["cause_stumbling"],
5    "specific_risks": []
6  },
7  {
8    "object": "tie",
9    "properties": [],
10   "specific_risks": [
11     {
12       "name": "asphyxiation",
13       "en": "If it is tied tightly, it can cause asphyxiation",
14       "es": "Si está atado con fuerza, puede causar asfisia"
15     }
16   ]
17 },
18 {
19   "object": "suitcase",
20   "properties": ["cause_stumbling"],
21   "specific_risks": []
22 },
23 {
24   "object": "frisbee",
25   "properties": ["break_objects"],
26   "specific_risks": []
27 },
28 {
29   "object": "bat",
30   "properties": ["cause_stumbling", "break_objects"],
31   "specific_risks": []
32 },
33 {
34   "object": "racket",
35   "properties": ["cause_stumbling", "break_objects"],
36   "specific_risks": []
37 },
38 {
39   "object": "skateboard",
40   "properties": ["cause_stumbling", "break_objects"],
41   "specific_risks": []
42 },
43 {
44   "object": "bottle",
45   "properties": ["liquid_container", "fragile"],
46   "specific_risks": []
47 },
48  ...
```

Figure B.2: Fragment of the objects file which comprises the Risks-objects corpus (Object 8-15).

```
1  ...
2  {
3    "object": "wineglass",
4    "properties": ["liquid_container", "fragile"],
5    "specific_risks": []
6  },
7  {
8    "object": "cup",
9    "properties": ["liquid_container", "fragile"],
10   "specific_risks": []
11  },
12  {
13   "object": "fork",
14   "properties": ["sharp", "choke"],
15   "specific_risks": []
16  },
17  {
18   "object": "knife",
19   "properties": ["sharp"],
20   "specific_risks": []
21  },
22  {
23   "object": "spoon",
24   "properties": ["choke"],
25   "specific_risks": []
26  },
27  {
28   "object": "bowl",
29   "properties": ["fragile"],
30   "specific_risks": []
31  },
32  {
33   "object": "banana",
34   "properties": ["eatable"],
35   "specific_risks": []
36  },
37  {
38   "object": "apple",
39   "properties": ["eatable"],
40   "specific_risks": []
41  },
42  {
43   "object": "sandwich",
44   "properties": ["eatable"],
45   "specific_risks": []
46  },
47  ...
```

Figure B.3: Fragment of the objects file which comprises the Risks-objects corpus (Object 16-24).

```
1  ...
2  {
3    "object": "orange",
4    "properties": ["eatable"],
5    "specific_risks": []
6  },
7  {
8    "object": "broccoli",
9    "properties": ["eatable"],
10   "specific_risks": []
11  },
12  {
13   "object": "carrot",
14   "properties": ["eatable"],
15   "specific_risks": []
16  },
17  {
18   "object": "hotdog",
19   "properties": ["eatable"],
20   "specific_risks": []
21  },
22  {
23   "object": "pizza",
24   "properties": ["eatable"],
25   "specific_risks": []
26  },
27  {
28   "object": "donut",
29   "properties": ["eatable"],
30   "specific_risks": []
31  },
32  {
33   "object": "cake",
34   "properties": ["eatable"],
35   "specific_risks": []
36  },
37  {
38   "object": "chair",
39   "properties": ["cause_stumbling"],
40   "specific_risks": [
41     {
42       "name": "breakable",
43       "en": "If a person sits on it and it is constructed from soft materials, it can ↔
44           ↔ break.",
45       "es": "Si una persona se sienta en él y está construido con materiales blandos, ↔
46           ↔ puede romperse."
47     }
48   ]
49  },
50  ...
```

Figure B.4: Fragment of the objects file which comprises the Risks-objects corpus (Object 24-32).

```
1  ...
2  {
3    "object": "couch",
4    "properties": ["cause_stumbling"],
5    "specific_risks": []
6  },
7  {
8    "object": "plant",
9    "properties": ["cause_stumbling", "fragile"],
10   "specific_risks": []
11  },
12  {
13   "object": "bed",
14   "properties": ["cause_stumbling"],
15   "specific_risks": []
16  },
17  {
18   "object": "table",
19   "properties": ["cause_stumbling"],
20   "specific_risks": []
21  },
22  {
23   "object": "toilet",
24   "properties": ["cause_stumbling", "flood"],
25   "specific_risks": []
26  },
27  {
28   "object": "tv",
29   "properties": ["electric", "energy_consuming"],
30   "specific_risks": []
31  },
32  {
33   "object": "laptop",
34   "properties": ["electric"],
35   "specific_risks": []
36  },
37  {
38   "object": "mouse",
39   "properties": ["electric"],
40   "specific_risks": []
41  },
42  {
43   "object": "remote",
44   "properties": ["electric"],
45   "specific_risks": []
46  },
47  ...
```

Figure B.5: Fragment of the objects file which comprises the Risks-objects corpus (Object 33-40).

```

1  ...
2  {
3    "object": "keyboard",
4    "properties": ["electric"],
5    "specific_risks": []
6  },
7  {
8    "object": "cellphone",
9    "properties": ["electric"],
10   "specific_risks": []
11  },
12  {
13   "object": "microwave",
14   "properties": ["electric", "hot_interior", "cause_burn_down"],
15   "specific_risks": []
16  },
17  {
18   "object": "oven",
19   "properties": ["electric", "hot_interior", "cause_burn_down"],
20   "specific_risks": []
21  },
22  {
23   "object": "toaster",
24   "properties": ["electric", "hot_interior", "cause_burn_down"],
25   "specific_risks": []
26  },
27  {
28   "object": "sink",
29   "properties": ["water_related"],
30   "specific_risks": []
31  },
32  {
33   "object": "refrigerator",
34   "properties": ["energy"],
35   "specific_risks": [
36     {
37       "name": "spoil_food",
38       "en": "If the refrigerator door remains open for a long time, it can cause the ↵
39         ↵ food inside to spoil.",
40       "es": "Si la puerta del refrigerador permanece abierta durante un período ↵
41         ↵ prolongado, puede provocar que los alimentos en su interior se ↵
42         ↵ deterioren."
43     }
44   ]
45  },
46  {
47   "object": "book",
48   "properties": ["cause_stumbling"],
49   "specific_risks": []
50  },
51  ...

```

Figure B.6: Fragment of the objects file which comprises the Risks-objects corpus (Object 41-48).


```

1  ...
2  {
3    "object": "clock",
4    "properties": ["electric", "fragile"],
5    "specific_risks": [
6      {
7        "name": "take_down",
8        "en": "If it is a wall clock, it can accidentally fall and cause injuries to ↔
9          ↔ people in its vicinity.",
10       "es": "Si se trata de un reloj de pared, puede caerse accidentalmente y causar ↔
11         ↔ lesiones a las personas que se encuentren cerca."
12       }
13     ],
14     {
15       "object": "vase",
16       "properties": ["liquid_container", "fragile"],
17       "specific_risks": []
18     },
19     {
20       "object": "scissors",
21       "properties": ["sharp"],
22       "specific_risks": []
23     },
24     {
25       "object": "teddybear",
26       "properties": ["cause_stumbling"],
27       "specific_risks": []
28     },
29     {
30       "object": "hairdryer",
31       "properties": ["electric"],
32       "specific_risks": [
33         {
34           "name": "burn",
35           "en": "If it is used in hot mode and pointed at a person's body for a few ↔
36             ↔ seconds, it can cause burns.",
37           "es": "Si se utiliza en modo caliente y se apunta al cuerpo de una persona ↔
38             ↔ durante unos segundos, puede provocar quemaduras."
39         }
40       ]
41     },
42     {
43       "object": "toothbrush",
44       "properties": ["choke"],
45       "specific_risks": []
46     }
47   ]
48 }

```

Figure B.7: Fragment of the objects file which comprises the Risks-objects corpus (Object 49-56).

B.2. Properties corpus

```
1  ...
2  {
3    "name": "eatable",
4    "risks": [
5      {
6        "name": "choke",
7        "en": "If it is eaten too fast or improperly, it can cause choking.",
8        "es": "Si se come demasiado rápido o de forma inadecuada, puede provocar asfixia↔
9          ↔ ."
10     }
11   ]
12 },
13 {
14   "name": "swallowable",
15   "risks": [
16     {
17       "name": "choke",
18       "en": "If it is used improperly, it can cause choking.",
19       "es": "Si no se utiliza correctamente, puede provocar atragantamientos."
20     }
21   ]
22 },
23 {
24   "name": "liquid_container",
25   "risks": [
26     {
27       "name": "spill_liquid",
28       "en": "If it collides with something, its content can be spilt.",
29       "es": "Si choca con algo, su contenido puede derramarse."
30     }
31   ]
32 },
33 {
34   "name": "sharp",
35   "risks": [
36     {
37       "name": "cause_cuts",
38       "en": "If it used improperly, it can cause injuries.",
39       "es": "Si no se utiliza correctamente, puede provocar lesiones."
40     }
41   ]
42   ...
```

Figure B.8: Fragment of the properties file which comprises the Risks-objects corpus (Property 1-4).

```
1  [
2    {
3      "name": "cause_stumbling",
4      "risks": [
5        {
6          "name": "collide",
7          "en": "If the object is placed in an unexpected location or if a person becomes ↵
8             ↵ confused, it can lead to accidental falls or injuries.",
9          "es": "Si el objeto está colocado en un lugar inesperado o si una persona se ↵
10             ↵ confunde, puede provocar caídas accidentales o lesiones."
11        }
12      ],
13    },
14    {
15      "name": "playable",
16      "risks": [
17        {
18          "name": "break_objects",
19          "en": "If it is used for playful purposes without exercising caution, it has the ↵
20             ↵ potential to collide with other objects, resulting in their breakage.",
21          "es": "Si se utiliza con fines lúdicos sin tener precaución, puede chocar con ↵
22             ↵ otros objetos que pueden romperse."
23        }
24      ],
25    },
26    {
27      "name": "alive",
28      "risks": [
29        {
30          "name": "attack",
31          "en": "If it becomes confused or nervous, it can attack causing injuries.",
32          "es": "Si se confunde o se pone nervioso, puede atacar causando lesiones."
33        }
34      ],
35    },
36    ...
37  ]
```

Figure B.9: Fragment of the properties file which comprises the Risks-objects corpus (Property 5-7).

```

1  ...
2  {
3    "name": "electric",
4    "risks": [
5      {
6        "name": "short-circuit",
7        "en": "If a short-circuit happens, it can cause electric shocks or burns.",
8        "es": "Si se produce un cortocircuito, puede causar descargas eléctricas o ↵
           ↵ quemaduras."
9      }
10   ]
11 },
12 {
13   "name": "energy_consuming",
14   "risks": [
15     {
16       "name": "energy",
17       "en": "If it is powered on for a long time, it can lead to high energy ↵
           ↵ consumption expenses.",
18       "es": "Si está encendido durante mucho tiempo, puede generar gastos elevados de ↵
           ↵ consumo de energía."
19     }
20   ]
21 },
22 {
23   "name": "hot_interior",
24   "risks": [
25     {
26       "name": "high_temperature",
27       "en": "If a person touches the interior of the object when it is powered on, ↵
           ↵ they can get burned.",
28       "es": "Si una persona toca el interior del objeto cuando está encendido, esta ↵
           ↵ puede quemarse."
29     }
30   ]
31 },
32 ...

```

Figure B.10: Fragment of the properties file which comprises the Risks-objects corpus (Property 8-11).

```
1  ...
2  {
3    "name": "fragile",
4    "risks": [
5      {
6        "name": "dangerous_pieces",
7        "en": "If it is made of a fragile material and collides with something, it can ↵
8        ↵ shatter into dangerous pieces that can cause injuries.",
9        "es": "Si está hecho de un material frágil y choca contra algo, puede romperse ↵
10       ↵ en pedazos peligrosos que pueden causar lesiones."
11     }
12   ],
13   {
14     "name": "water_related",
15     "risks": [
16       {
17         "name": "flood",
18         "en": "If water accumulates beyond the available capacity, it can cause flooding↵
19         ↵ .",
20         "es": "Si el agua se acumula por encima de la capacidad disponible, puede ↵
21         ↵ provocar inundaciones."
22       }
23     ]
24   },
25   {
26     "name": "cause_burn_down",
27     "risks": [
28       {
29         "name": "burn_object",
30         "en": "If a flammable object is put on its interior, it can burn down",
31         "es": "Si se coloca un objeto inflamable en su interior, este puede arder"
32       }
33     ]
34   }
35 ]
```

Figure B.11: Fragment of the properties file which comprises the Risks-objects corpus (Property 12-14).

C. Appendix III



Annotation: Sandwich, table

Detections: Refrigerator, bottle, cup, bowl, chair

Figure C.1: This figure represents a demonstration of the low amount of object annotations provided by Charades. Although a rich set of objects is detected, none of these are in the annotations set so the Recall score for the Object Recognition task in this video is 0,0.



Figure C.2: This figure represents a demonstration of how poor quality videos affects Objects Recognition results. This frame shows a person drinking a cup of coffee. As the video has poor quality both SwinBERT and YOLOv7 are not able to recognize any object (only recognizes the person). The annotations objects set of this video is only composed by the object "Cup".