

# A Novel Multimodal Approach for Surgery Video Report Generation with Large Language Models

Hugo Georgenthum

March 15, 2025



Master's Thesis

**Supervisors:** Mishra Siddhartha, Pietro Liò, Cristian Cosentino

## Abstract

The automatic summarization of surgical videos is crucial for improving procedural documentation, surgical training, and post-operative analysis. This thesis presents a new method at the intersection of artificial intelligence and medicine, seeking to develop innovative machine-learning models with real-world applications in surgery. To this end, we propose a multi-modal approach to generate video summaries by benefiting from the latest improvements in both computer vision and large language models. For instance, the model processes surgical videos in the 3 following key steps. After dividing the video into clips, the focus is on the extraction of visual features, by treating the clips on a frame level with visual transformers. The goal is to detect the tools, organs, tissues and actions performed by the surgeon. These visual features are then translated to frame captions using large language models. Subsequently, on the video level, the emphasis is placed on the temporal features. The latter are obtained with a Vivit-based encoder by taking as input both the clips and the frame captions extracted earlier. In an analogous way to the frame captions, the temporal features are converted into clip captions, which capture the overall context of the clip. The last phase gathers the combination of the clip descriptions into a surgical report with an LLM specifically designed for this task. We train and evaluate our model on the CholecT50 dataset, leveraging instrument and action frame annotations along 50 laparoscopic videos. Experimental results demonstrate that our method produces coherent and contextually meaningful summaries, with a 96% precision for tool detection and 0.74% Bert score for temporal context extraction. This research contributes to the development of AI-assisted tools for surgical reporting and analysis.

# Acknowledgments

Many thanks to **M. Mishra Siddhartha** for his support and making this thesis possible.

Un ringraziamento speciale a **Cristian Cosentini** e **Pietro Liò** per la loro supervisione e il loro aiuto in questo progetto gratificante.

A **mes parents** et ma soeur **Louise** pour leur encouragements à tous les égards tout au long de mes études.

شكرا بزاف لخديجتي ديايلى سليمى على المساعدة والنصائح ديايلى فوقت الصعبة. وزيد،  
كاتبان بحال راكون.

Moltes gràcies als meus amics **Mar** i **Marco** per aixecar la meua moral perdent cada dia contra mi al billar.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Literature Review</b>	<b>7</b>
2.1	Video to Text Models . . . . .	7
2.2	Computer vision in Surgery . . . . .	8
<b>3</b>	<b>Background</b>	<b>10</b>
3.1	Fine-tuning . . . . .	10
3.2	Attention mechanisms . . . . .	10
3.3	Transformers . . . . .	11
3.4	Vision Transformer . . . . .	13
3.5	Large Language Models . . . . .	14
<b>4</b>	<b>Methodology</b>	<b>16</b>
4.1	Object Detection . . . . .	17
4.1.1	Vision Transformer for Multi-Label Classification . . . . .	17
4.1.2	Weighted Loss Function . . . . .	18
4.2	Frame Caption Generation . . . . .	18
4.3	Clip caption Generation . . . . .	19
4.4	Final Summary Generation . . . . .	20
<b>5</b>	<b>Practical Setting</b>	<b>21</b>
5.1	Dataset . . . . .	21
5.2	Pre-trained Models . . . . .	22
5.2.1	Clip caption generation scheme . . . . .	24
5.3	Evaluation Metrics . . . . .	26
5.3.1	Classification metrics . . . . .	26
5.3.2	Calibration . . . . .	26
5.3.3	Text Metrics . . . . .	27
5.4	Training Strategy . . . . .	29
5.5	Resources and technical specifications . . . . .	29
<b>6</b>	<b>Experiments</b>	<b>30</b>
6.1	Workflow Example . . . . .	30
6.2	Results . . . . .	31
6.2.1	Object Detection . . . . .	31
6.2.2	Frame Caption Generation . . . . .	32
6.2.3	Clip caption Generation . . . . .	32



# Chapter 1

## Introduction

Artificial intelligence (AI) is increasingly transforming the field of surgery by enhancing precision, efficiency, and decision-making. AI-based systems can be used to assist surgeons by providing real-time guidance, automating routine tasks, and analyzing vast amounts of surgical data [Hashimoto et al., 2018]. These advancements reduce human error and aims to improve patient safety, making AI a more and more valuable tool in modern surgical procedures.

One of the key developments in AI is computer vision, which has significantly improved in recent years. Computer vision describes the methods designed to analyze and interpret complex visual data. A fundamental tool in this domain is Convolutional neural networks (CNNs), which achieves remarkable results in image processing. These deep neural networks proved to be effective for a wide range of tasks, such as image-based diagnostics and surgical assistance in medicine, thanks to their ability to learn spacial hierarchies in images. However, one key challenge not addressed by CNNs is their limited ability to capture long-range dependencies. In computer vision, the emergence of transformers, a powerful neural network architecture, marked a major breakthrough by overcoming this limitation. Unlike CNNs, transformers adopt self-attention mechanisms allowing them to model complex spatial relations more effectively (which will be explained later), and thus, surpassing CNNs in many different applications.

Another domain greatly impacted by transformers is Natural language processing, as it became an important technique for processing text datasets. Indeed, large language models (LLMs), famously known after the release of ChatGPT, can understand and generate human-like text thanks to transformers. To capture complex language patterns and contextual relationships, the LLMS are trained on vast amounts of textual data. As a result, these models can perform a wide range of language tasks, including text generation and summarization, with remarkable accuracy.

In this thesis, we use recent advancements in computer vision and natural language processing to use the large amounts of surgical videos available in the world and extract useful information. Having this in mind, we combine these approaches to automatically generate surgical reports from video data. By analyzing surgical videos, AI systems can detect key instruments, actions, and critical events in real-time, creating structured, detailed summaries. The deep analysis offered by our model will improve our surgical documentation and help surgeons avoid errors and unnoticed anomalies. Thus, the deployment of such a

tool in real-time surgeries holds great potential for improving surgical practices, offering valuable insights, and contributing to better patient care.

Integrating AI into surgery comes with significant challenges. Surgical procedures require extreme precision, where even small mistakes can have serious consequences. AI models must be highly reliable to support surgeons in making critical decisions. In addition, these systems need to be explainable, offering clear argumentation for their recommendations for the surgeons to trust the model. They also need to be adaptable to different surgical environments, as procedures can vary greatly depending on the surgery type. Finally, it is also important to say that integrating AI presents ethical challenges as well. Data sensitivity presents many issues and patients should always consent. As many machine learning models, biases in the training data can introduce bias in the outcome and thus put minority populations at harm. Finally, we recall that AI must remain an assistive tool for the surgeons rather than a tool to make decisions.

This thesis is organized as follows. Chapter 2 covers the latest literature in the domain, highlighting key studies and advancements relevant to this work, while chapter 3 describes the necessary scientific background to understand the methods. Chapters 4 and 5 respectfully describe the theoretical and practical implementation of the methodology. Finally, the results can be found in chapter 6.

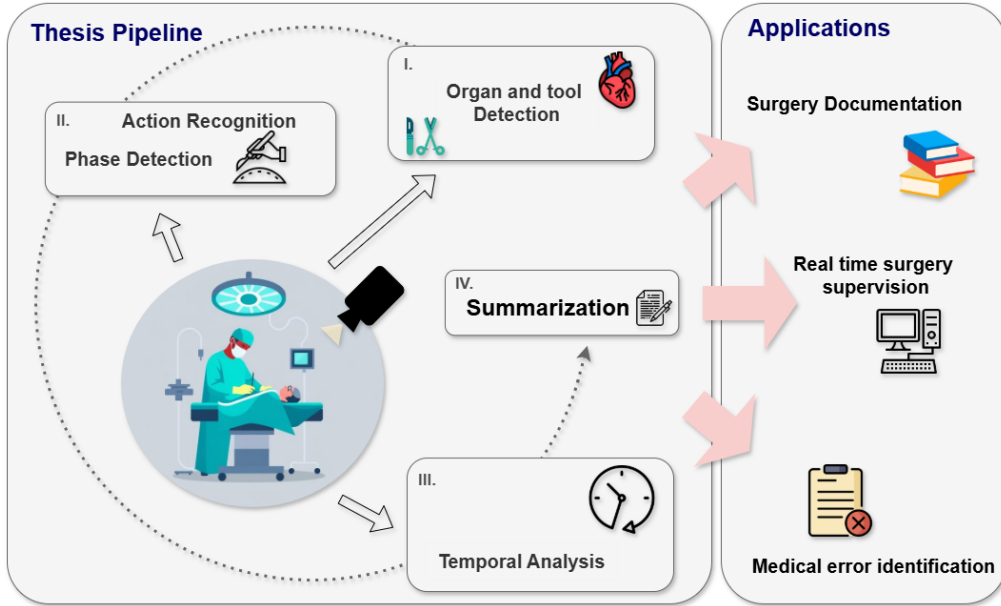


Figure 1.1: Overview of different modalities in surgical video and applications.

## Chapter 2

# Literature Review

### 2.1 Video to Text Models

One major issue encountered by traditional video models is information loss. Indeed, processing frame as a sequence of independent item leads to information loss and inefficient representations. To address this, Feng et al. [2024] develop the model *VideoOrion* that presents a novel way of understanding videos by focusing on both the overall scene and individual objects. *VideoOrion* introduces a two-branch approach: (i) an object-centric branch and (ii) a video-centric branch. Precisely, (i) the object-centric branch is designed to track and analyze objects separately. Using a detect-segment-track pipeline, this branch isolates objects and converts them into object tokens, which represent their spatial features. This approach significantly improves the model’s ability to recognize object interactions, track movements more precisely, and reduce redundancy in video processing. By focusing on objects rather than entire frames on a first time, it retains essential information while omitting irrelevant background details, making it both more efficient and more interpretable. On the other hand, (ii) the video-centric branch captures global scene context by processing frame-level features. This ensures that the overall structure of the video is preserved, allowing the model to understand relationships between objects and background elements. The combination of both branches allows *VideoOrion* to maintain a balance between object details and global scene understanding. The results demonstrate that *VideoOrion* significantly enhances performance on video question-answering and understanding tasks. By incorporating a specialized object detection branch and integrating its outputs into the LLM, *VideoOrion* provides additional structured information, helping the model maintain accuracy, avoid hallucinations, and ensure no objects are overlooked.

Zhao et al. [2024] develop a similar model *VideoPrism* as their model uses a temporal attention module, with two layers. The first layer captures spatial information within each frame, whereas the second layer models how frames relate to each other over time. This dynamic allows the model to learn how actions and events unfold across frames. Also, an important technique used in *VideoPrism* is that the videos are divided into manageable segments, which is defined by temporal tokenization. This tokenization allows the model to focus on both individual frames and relationships between them. This approach enhances the model’s ability to treat longer videos, which is crucial for understanding how



actions evolve over a long time.

The paper presenting the model *VideoGLaMM* [Munasinghe et al., 2025] builds on the idea of improving video understanding by focusing on both spatial details (where things are) and temporal dynamics (how they move or change over time). While *VideoOrion* separates object-level and scene-level information, *VideoGLaMM* takes this a step further by clearly distinguishing positional and temporal features. *VideoGLaMM* is designed with three main parts: a large language model (LLM), two separate vision encoders, and a decoder. Having two separate vision encoders enables to separate the processing of the frames. One encoder focuses on analyzing individual frames, capturing spatial details such as object positions and appearances. The second encoder looks at the video as a whole, tracking how objects move and change over time. By keeping these two aspects separate, it allows the model to better understand what is happening during a scene at any given moment and how actions and plans evolves through the video. This makes it more accurate when matching text descriptions to specific objects and moments. To ensure smooth processing in the model, the authors include special adapters that help combine frame-based and video-wide information effectively. Finally, to show the performances of the model, they created a massive dataset with 38,000 video-based questions, 83,000 objects, and over 671,000 segmentation masks for training and testing *VideoGLaMM*, thus allowing the model to learn fine-grained video grounding. The results outline that *VideoGLaMM* outperforms previous models on tasks such as identifying objects in videos based on text prompts and generating grounded responses in conversations.

One major challenge in Large Vision-Language Models is that models can misinterpret or imagine details that are not present in the given video, referred to as hallucinations. Xu et al. [2023] address this problem by proposing a multi-turn reasoning framework that helps reduce hallucinations by encouraging models to refine their understanding step by step. This technique allows the models to revisit their previous outputs, to verify the consistency with provided visual input, and correct potential errors.

One key point to retain from this section is that it is essential to divide into several heads when analyzing videos. We have seen that the ability to separate spatial and temporal information proves to be a game-changer, as it helps the model understand complex video content with greater precision. However, improvement is still possible in this direction. In this thesis, we develop a similar approach that combines spatial interpretation on objects (*VideoOrion*) and and others on overall context (*VideoGLaMM*). By considering both possibilities at the same time, we aim to strengthen the model’s ability to capture visual information.

## 2.2 Computer vision in Surgery

Image analysis has been a huge improvement in the realm of medicine, in particular in oncology. AI proved to be very efficient to detect anomalies in MRI, CT scans or other imagery and sometimes earlier than traditional methods. However, no video-to-text models have been developed yet. Computer vision models became more popular for different domains of medicine, including surgery. In

particular, video recognition AI has emerged and shown promising results. For example, Nasirihaghighi et al. [2023] focuses on the possible applications of computer vision on laparoscopic videos, in particular in gynecology. The aim of the presented model is to use a hybrid transformer model to recognize key events during the surgery, by capturing long-range dependencies across video frames. This is particularly important when it comes to surgery videos, since actions can be very long. Due to the length of actions, the model struggles to detect the procedures performed by the surgeon. The hybrid approach combines the strength of transformers with other architectures, making it possible for the model to deal with other challenges specific to laparoscopic videos. Indeed, the laparoscope operates in tight spaces inside the human body, often causing occlusion and motion blur. The laparoscopic videos utilized for this project are annotated with critical events, like major complications or significant surgical steps. This approach, by focusing on inter-frame dependencies, outperforms traditional models like CNN-RNN in recognizing complex events in laparoscopic videos.

Similarly, Kiyasseh et al. [2023] develops a Vision Transformer (ViT) model designed to analyze surgical activities from video data. The model distinguishes itself by its integration of a supervised contrastive learning framework, which improves feature discrimination by encouraging the model to learn distinct representations for different surgical activities. By introducing a loss function that computes the similarity between frames, it allows the model better differentiate between variations in surgical actions. This approach is particularly valuable thanks to its generalization across a variety of surgical procedures, making it adaptable to different contexts.

In the framework of surgery videos, access to data with qualitative annotations is still relatively restrained, making the analysis challenging. Whether it is for economic or logistical reasons, it can be difficult for hospitals to provide the necessary dataset with the corresponding annotations. Studies find ways to address this problem but using coresets, for example [Volkov et al., 2017]. A Coreset is a subset of a larger dataset, that approximates the key informations from the original one. The aim is to be able to train the models more efficiently on a smaller dataset, while maintaining the accuracy. The dataset reduction, coupled with more computationally efficient models than vision transformers, in particular support vector machine (SVM), allows the model to generate a reliable real-time feedback to the surgeons. Indeed, vision transformers are heavy models in terms of number of parameters, making a real-time inference impossible for some cases. The paper explores an interesting strategy for video analysis efficiency, in particular towards efficiency and instantaneous supervision during procedures.

## Chapter 3

# Background

This chapter covers the essential mathematical concepts and machine learning techniques needed to fully understand the methods introduced in this thesis.

### 3.1 Fine-tuning

Fine-tuning a neural network involves optimizing a pre-trained model on a new dataset by adjusting its parameters  $\theta$ . Given a dataset  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ , where  $x_i$  represents the input features and  $y_i$  the target labels, fine-tuning updates the model weights  $\theta$  to minimize a loss function  $\mathcal{L}(\theta)$ .

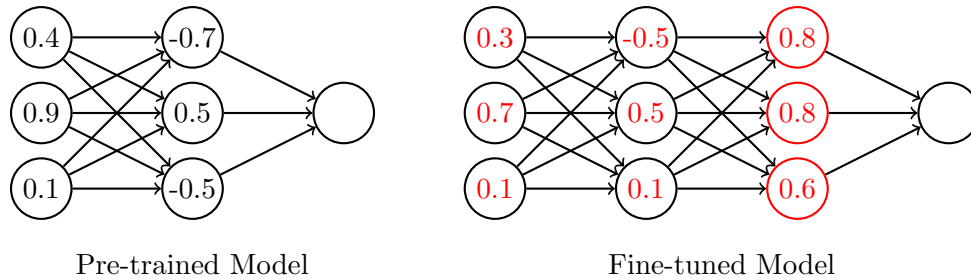


Figure 3.1: Fine-tuning a neural network by adding a new layer and updating node weights.

### 3.2 Attention mechanisms

Attention mechanisms are a method to help a model to focus on the most important part of an input sequence (e.g a list of tokens). While Recurrent neural networks use a fix window among the previous tokens to generate the next one, Attention mechanisms attach an importance to each of the tokens to help the model better understand the dependencies between them. This importance is translated into weights, which aim to retain the most important information. This technique is particularly useful when it comes to input sequences with

long-range dependencies. Let  $X \in \mathbb{R}^{n \times d}$  be a sequential dataset. Three vectors are associated to the input: query (Q), key(K), and value (V). The Query represents which tokens are important for each one. The key corresponds to the information retained in each token. Finally, the Value, which will be used for prediction, denotes the important information from each token. The formulas are as follows:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V$$

Where  $W^Q$ ,  $W^K$ ,  $W^V$  being respectively in  $\mathbb{R}^{d \times d_k}$ ,  $\mathbb{R}^{d \times d_k}$ ,  $\mathbb{R}^{d \times d_v}$ , are learnable weights matrices updated during the training. We get the following expression for a single attention head:

$$\text{Attention head}(Q, K, V) = \underbrace{\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)}_{A := \text{Attention weights}} \cdot V := \text{softmax}(S) \cdot V = A \cdot V$$

with  $d_k = n/h$  with  $h$  the number of attention head. Intuitively, the "importance"  $QK^T$  is scaled by the dimension  $d_k$ , and then transformed into probabilities using the softmax function. The heads are then concatenated into a multiple head:

$$\text{Multiple head} = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O$$

with  $W^O$  an other learnable weights matrices. The benefit of using multiple heads is to be able to capture different kinds of relationships and dependencies among the tokens.

In some cases, it is important to ensure tokens only consider the previous ones to prevent information leakage. To this aim, masks can be applied to avoid attention scores to consider the dependencies with future tokens. The weights are updated as such:  $A' = \text{softmax}(S')$ , with

$$S' = S + M, \quad S', S, M \in \mathbb{R}^{n \times n} \quad \text{and} \quad M_{ij} = \begin{cases} 0 & \text{if } j \leq i \\ -\infty & \text{otherwise} \end{cases}$$

The  $-\infty$  ensures a 0 probability will be attributed to the corresponding entry after using softmax.

### 3.3 Transformers

The Transformer architecture, introduced for the first time in the state-of-the-art paper *Attention is all you need* Vaswani et al. [2023] leverages attention mechanisms within an encoder-decoder framework. The overall architecture can be seen in 3.2.

1. **Encoder:** The encoder consists of a stack of  $N$  identical layers. Each encoder layer applies multi-head self-attention and a positional feed-forward network. For an input  $X$  (including positional encodings), one encoder layer is expressed as:

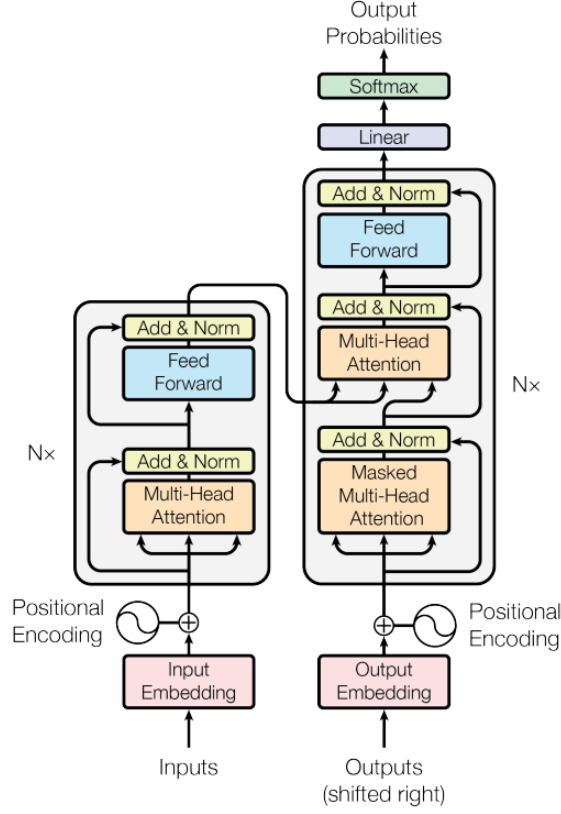


Figure 3.2: Transformer architecture

$$\begin{aligned}\tilde{X} &= \text{LayerNorm}\left(X + \text{MultiHead}(X, X, X)\right), \\ X' &= \text{LayerNorm}\left(\tilde{X} + \text{FFN}(\tilde{X})\right).\end{aligned}$$

After encoding  $N$  layers, the encoder outputs:

$$E = \text{EncoderLayer}^N(X).$$

## 2. Decoder:

The decoder is also composed of  $N$  layers, each of them having three sub-layers:

Masked Multi-Head Self-Attention to prevent future token information leakage as explained earlier. Encoder-Decoder Attention focuses on the encoder output  $E$  to help the decoder generate predictions. Feed-Forward Network For a decoder input  $Y$  (shifted right during training), a single decoder layer is formulated as:

$$\begin{aligned}\tilde{Y} &= \text{LayerNorm}\left(Y + \text{MaskedMultiHead}(Y, Y, Y)\right), \\ \hat{Y} &= \text{LayerNorm}\left(\tilde{Y} + \text{MultiHead}(\tilde{Y}, E, E)\right), \\ Y' &= \text{LayerNorm}\left(\hat{Y} + \text{FFN}(\hat{Y})\right).\end{aligned}$$

Stacking L decoder layers produces the final decoder output:

$$D = \text{DecoderLayer}^N(Y).$$

**3. Final Output:** The decoder output D is projected to the vocabulary space and converted to probabilities via softmax:

$$\hat{y} = \text{softmax}\left(W^O \cdot D + b\right).$$

In terms of performance, transformers outperform most Recurrent Neural Network (RNN) architectures. This difference of architecture allows to treat longer sequential inputs by avoiding exploding gradients. Indeed, augmenting the temporal window of RNN exposes them to exploding gradients, which is not the case for transformers.

### 3.4 Vision Transformer

Vision transformers (ViT) aim to adapt the transformer architecture for image input. introduced by Dosovitskiy et al. in "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale" Dosovitskiy et al. [2021], ViT models treat images as a sequence of "patches" instead of using convolutional layers like in CNNs. Thanks to the positional encoding, the model still captures the spatial dependencies between patches. The overall architecture presented in the paper can be seen in 3.3.

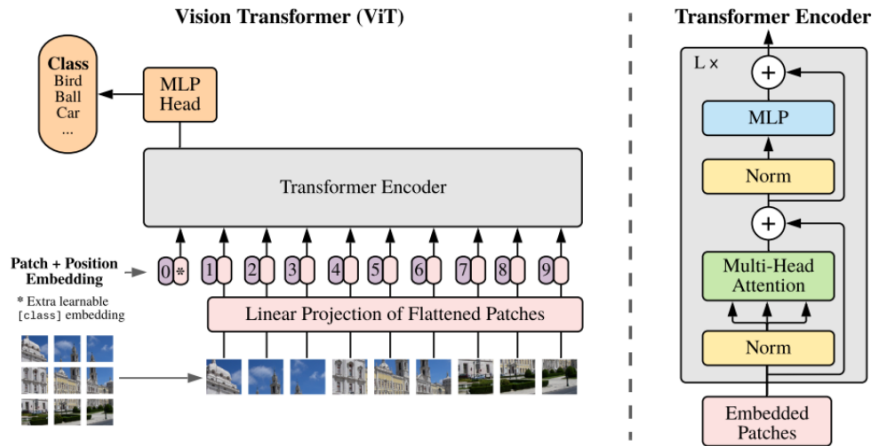


Figure 3.3: Transformer architecture

Each image is divided into fixed-size patches (e.g.,  $16 \times 16$  pixels), which are flattened and projected into a lower-dimensional embedding space via a linear layer. These patch embeddings are then processed by a standard transformer encoder. The paper demonstrates that ViT outperforms state-of-the-art CNNs when trained on large-scale datasets. In ImageNet-1k, ViT achieved excellent accuracy of 88.36% after fine-tuning, surpassing the ResNet-based model. It is highlighted that transformers are suitable to image processing when trained with sufficient data, marking a significant shift in computer vision architecture design.

### 3.5 Large Language Models

A Large Language Model (LLM) is a deep learning model trained on vast amounts of text data to understand and generate human-like text. They process input text in the form of *tokens*, i.e. sub-word units obtained through a process called tokenization. The workflow of an LLM can be divided into three stages: encoding, transformation, and decoding.

1. When text input is provided to an LLM, it is first tokenized into discrete tokens. These tokens are then embedded to a latent space.
2. During the transformation stage, the embedded tokens are processed through a sequence of layers. This stage is crucial for understanding relationships between words and forming a coherent latent representation of the input text.
3. In the decoding stage, the model generates its output text by predicting the next token iteratively. The decoding process starts with an initial token (often a special token like [BOS] for beginning-of-sequence). The model generates the next token based on the encoded representations and the previously generated tokens. This process continues until a stopping criterion, most of the time by reaching a maximum length or generating an end-of-sequence token.
4. To improve the quality of generated text, there is different decoding parameters that balances diversity and reliability.
  - (a) Temperature ( $T$ ): This parameter modifies the logits before applying softmax, scaling them as  $p_i \propto e^{z_i/T}$ . Lower values of  $T$  make the model more confident in its predictions, while higher values encourage more exploration.
  - (b) Top-k sampling: At each step, only the  $k$  most probable tokens are considered. This ensures that rare and low-probability words are not considered and keeps the output more predictable.
  - (c) Top-p (nucleus) sampling: Instead of a fixed  $k$ , the smallest set of tokens such that the cumulative probability exceeds  $p$  is selected,  $p$  being a threshold previously defined. This method makes sure that only the most relevant possible tokens are considered while allowing controlled variation.

- (d) No-repeat n-gram constraint: This is particularly useful to prevent the model from repeating phrases by ensuring that no n-gram (sequence of  $n$  words) appears more than once, improving fluency and coherence.

Since our task involves surgical video captioning, low randomness in generation should be prioritized to ensure that outputs remain accurate, reliable, and clinically meaningful.



## Chapter 4

# Methodology

Now that the necessary background has been presented, we can introduce the methodology of our model, which will be divided into 4 parts:

- Object detection
- Frame Caption generation
- Clip caption generation
- Report creation

Considering the challenges of rigorous precision needed in the realm of surgery video analysis, we present the following architecture to generate video reports 4.1. We consider a video  $V$  of  $N$  frames and a clip length  $l = 32$  frames. The video is divided into  $n = \lceil N/l \rceil$  clips:  $V = \{C_1, C_2, \dots, C_n\}$ .

Depending on the type of surgery, the procedure can be very long, up to a couple hours for some cases. To deal with this high temporal complexity, we propose a strategy of dividing videos into temporal tokens, more precisely into 32 seconds clips. By segmenting the video in this way, we enable the model to focus on more manageable chunks of temporal information, and create a more reliable clip description. Then, a complete report is created from the concatenation of the clip captions using an LLM. All the context information being already in the captions, the LLM focuses on making these captions into a more readable task, limiting this task to a simple summarization task.

$$\text{Summary} = \text{LLM}\left(\{\text{clip caption}_i\}_{i=1}^n\right)$$

The clip captions are generated using three distinct models. First, the object detection model identifies tools, organs, and other visual elements present in each frame. Next, a second model generates a frame caption by combining the detected objects with the frame itself. By treating object tokens as an additional modality, this model improves its efficiency at capturing the surgeon's actions. At this stage, processing videos as a collection of independent frames helps the model focus on spatial relationships in the frames themselves, identifying key objects and actions within each frame. Finally, we analyze entire clips to capture temporal dependencies between frames. The frame captions serve as a second modality, guiding the model to use the correct objects and actions in the final clip caption.

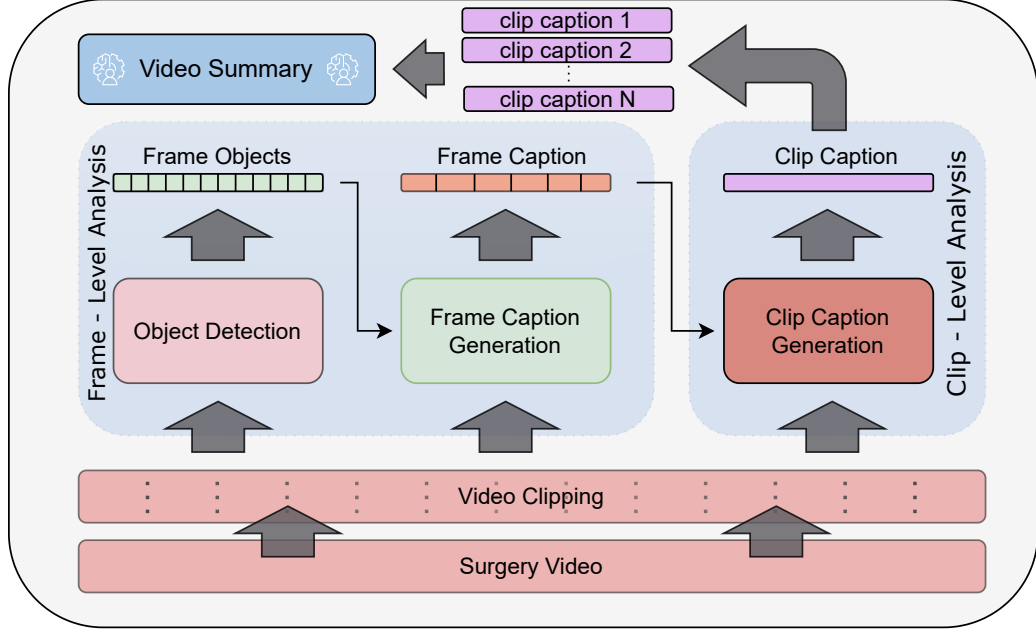


Figure 4.1: Model Architecture

## 4.1 Object Detection

The first module used in our approach is a vision transformer adapted to multi-label classification. The goal is to be able to detect all the present objects (e.g. tools, organs, tissues,...) among the ones known by the model. The input is the frame itself and returns a binary vector, each coordinate corresponding to a specific item.

### 4.1.1 Vision Transformer for Multi-Label Classification

The input images are transformed into  $[3, 224, 224] := [3, H, W]$  tensors, the three 3 corresponding to the RGB coordinates, and fed into the ViT which divides it into  $N$  non-overlapping patches of size  $16 \times 16$ .

Given an input image of dimensions, the Vision Transformer first divides it into non-overlapping patches of size, resulting in patches where:

$$N = \frac{H \times W}{p^2} = 196. \quad (4.1)$$

**Definition 4.1.1** For a matrix  $A = (a_{i,j}) \in \mathbb{R}^{n \times m}$ ,  $1 \leq i \leq i' \leq n$ ,  $1 \leq j \leq j' \leq m$ , we write

$$A_{[i:i',j:j']} = \begin{bmatrix} a_{i,j} & \dots & a_{i,j'} \\ \vdots & \ddots & \vdots \\ a_{i',j} & \dots & a_{i',j'} \end{bmatrix}$$

**Definition 4.1.2** Let  $A \in \mathbb{R}^{n \times m}$  and patch size  $a \times b$  with  $a, b$  divisible by  $n$ ,

$m$  respectively. We define a patch  $P_k \in \mathbb{R}^{a,b}$  with  $k = i * k + j$  as

$$P_k = A_{[i:i+1, j:j+1]}$$

and the flattened version of  $P_k$  is written  $\tilde{P}_k$

Now that we correctly defined our patches, and our ViT neural network returning a vector of the same as the number of classes, we can define the input format of an image  $I$  considering the positional encoding:  $I = [[1, \tilde{P}_1], [2, \tilde{P}_2], \dots, [1, \tilde{P}_N]] \in \mathbb{R}^{1 \times (16*16+1)*196}$ .

And we get

$$P = softmax(ViT(I))$$

We finally determine the present objects as a list  $O$  using a threshold such as  $O = \{\text{object}_i | P_i > \text{threshold}\}$ . A first reasonable choice for the threshold would be 0.5, which could be updated later depending on the calibration of the model.

#### 4.1.2 Weighted Loss Function

Due to the imbalance in object frequencies during surgeries, we employ a weighted binary cross-entropy loss function. Given the ground-truth labels and the predicted logits, the Binary Cross-Entropy Loss is defined as:

$$\mathcal{Loss} = - \sum_{i=1}^n w_i [y_i \log \sigma(\hat{y}_i) + (1 - y_i) \log(1 - \sigma(\hat{y}_i))]. \quad (4.2)$$

The weight for each class is computed based on the inverse of its frequency in the dataset:

$$w_i = \frac{1}{f_i + \epsilon}, \quad (4.3)$$

where  $f_i$  represents the number of occurrences of class  $i$  and  $\epsilon$  is a small constant to prevent division by zero. The weights are then normalized to sum to one.

## 4.2 Frame Caption Generation

Once the visual entities are detected, there is still spatial information to extract within the frames, such as the action being currently performed by the surgeon. The Frame caption generation model is designed to generate captions for video frames by leveraging both this remaining visual and the object detected earlier. This model integrates three main components: a ViT as seen before for frame encoding, a text encoder to encode the objects present within the surgical field, and a text decoder for generating captions. The combination of visual and textual representations enhances the quality of generated descriptions by incorporating contextual information. The same number of patches as the ViT for object detection 4.1 is used here.

The key components of the model are detailed here:

- **Frame Encoder:** A pre-trained ViT encodes input frames into a latent representation. The output features are then projected to a 512-dimensional space using a linear layer.
- **Object Encoder:** A text encoder  $E$  processes object labels associated with the frame. These textual features are also projected into a 512-dimensional space.
- **Fusion:** Both encodings are fused into a common latent space.
- **Text Decoder:** A pre-trained model decoder  $D$  generates captions based on the combined representations of the frame and object features.

Given an input Image  $I$  and a corresponding set of object labels  $O$ , the model generates a caption  $C$ . The processing steps are as follows:

$$\begin{aligned}
h_I &= \text{ViT}(I) && \in \mathbb{R}^{N \times 768} \\
h'_I &= W_I h_I && \in \mathbb{R}^{N \times 512} \\
h_O &= E(O) && \in \mathbb{R}^{S \times 768} \\
h'_O &= W_O h_O && \in \mathbb{R}^{S \times 512} \\
h_C &= \text{concat}(h'_I, h'_O) && \in \mathbb{R}^{(N+S) \times 512} \\
h'_C &= \text{CrossAttention}(h_C) \\
C &= D(h'_C)
\end{aligned}$$

where  $W_F$  and  $W_O$  are learnable projection matrices for the ViT and E outputs, respectively,  $N$  the same number of patches as in section (1), and  $S$  the number of tokens needed to encode  $O$ . The combined representation  $h_C$  is then passed to the decoder for caption generation.

### 4.3 Clip caption Generation

Now that the previous models captured the frame-level informations, we employ ourselves to capture the temporal dependencies by treating the input at a video level. A similar strategy is used as in the Frame Caption Generation, except for the following key points:

1. While the modalities remain the same (tensor and text), the dimensions change. The model now considers both the clips ( dimension  $[N_f, 3, H, W]$  with  $N_f$  number of frames in a clip), and the  $N_f$  frame captions of the corresponding frames.
2. A temporal dimension is added to the patches as seen in figure 4.2.

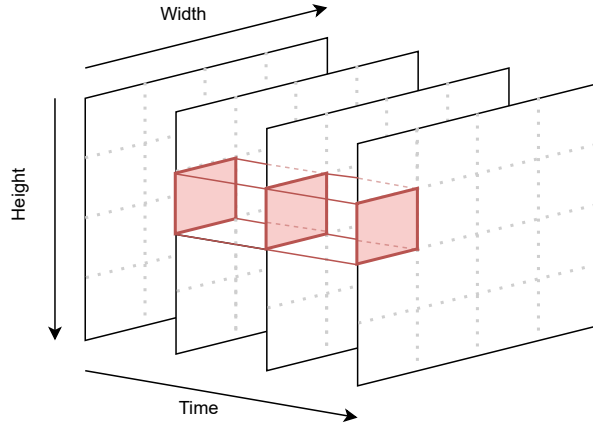


Figure 4.2: Video vision transformer patch example

## 4.4 Final Summary Generation

After generating clip captions for each segment of the video, the next step is to compile them into a cohesive final report. To achieve this, we use a LLM specifically designed for summarization. The model takes as input the complete set of clip captions along with a prompt instructing it to generate a structured and comprehensive report. This approach ensures that the final summary captures the most relevant details while making it more well-structured as a report. The chosen prompt for the LLM is explained later in section 5.2.

## Chapter 5

# Practical Setting

Once the global methodology is established, many practical aspects need to be defined. In this section, we present the dataset as well as the chosen models to be fine-tuned during the implementation. We also present the metrics used to evaluate these models.

### 5.1 Dataset

The CholecT50 CAMMA [2020] dataset used for the project has been provided by the University of Strasbourg and is composed of laparoscopic videos. A laparoscopic surgery is a procedure using a laparoscope, i.e. a thin tube equipped with a camera and a light. This kind of surgery is a subgroup of endoscopic procedures, often used for diagnostic purposes. The surgery in question is a *cholecystectomy*, aiming to remove the gallbladder, involving small incisions and the use of a camera. This operation typically occurs because of the presence of gallstones which may cause pain or infections.

The dataset is provided as 50 video folders, each video having a rate of 1 frame per second (fps), as well as tool, action, target and phase label for every frame. In total, the dataset encompasses 100 distinct triplet categories derived from 6 instruments, 10 verbs, and 15 targets, resulting in over 151,000 annotated triplet instances. With a total of 89827, the clips are created by grouping the frames by 32, with an overlap of 16 frames between every clip, resulting in 6232 clips.

Frame caption and clip captions are not provided in the original dataset, and are created artificially using the annotations. To create the frame captions, the verb, target and phase are used to construct a simple sentence presenting what the surgeon is doing in the frame. The same goes for the clip caption construction; the frame captions are concatenated into one clip captions, taking into account the time and order each action is taken during the clip. An example can be seen in 5.1.



Figure 5.1: Frame example: video = *VID01*, frame = 000000.png

**Detected objects** : grasper, gallbladder

**Frame caption** : During phase preparation, the grasper is grasping the gallbladder

**Corresponding clip caption** : First, during the phase of preparation lasting 22 seconds, the grasper is grasping the gallbladder while the hook is present. Then, during the phase of carlot-triangle-dissection lasting 10 seconds, the grasper is grasping the gallbladder while the hook is present.

A *cholecystectomy* is divided in 6 key phases. The number of frames corresponding to each of these phases is available in table 5.1.

Phase Name	Frame Count	Time (minutes)
Preparation	2806	46.8
Calot-triangle-dissection	38808	646.8
Clipping-and-cutting	7790	129.8
Gallbladder-dissection	26789	446.5
Gallbladder-packaging	3790	63.2
Cleaning-and-coagulation	6986	116.4
Gallbladder-extraction	2858	47.6
<b>Total</b>	<b>89927</b>	<b>1498.8</b>

Table 5.1: Duration of Surgical Phases ( $fps = 1$ )

As previously mentioned, objects in the dataset exhibit a significant variation in occurrence frequency, with some appearing far more often than others. This imbalance is illustrated in Figure 5.2.

## 5.2 Pre-trained Models

To effectively capture and summarize surgical video content, we leverage multiple pre-trained models, each chosen for its specific strengths in processing visual and textual information.

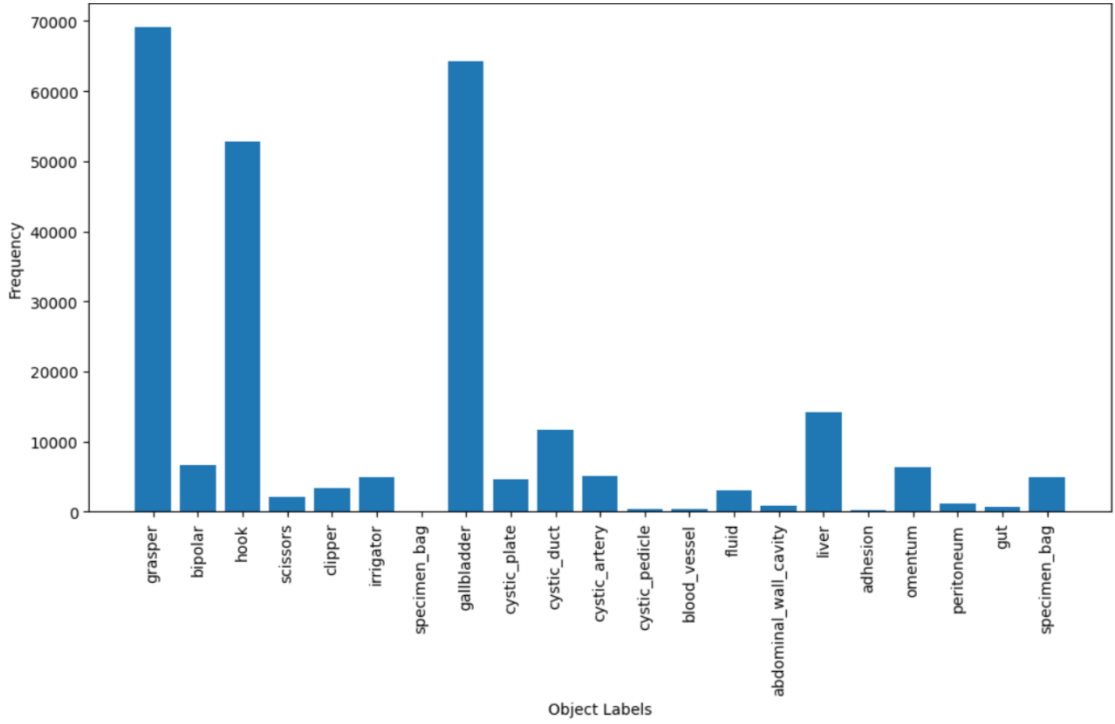


Figure 5.2: Object Occurencies in the dataset CholecT50

For encoding individual frames and video sequences, we employ two transformer-based vision models:

- ViT (Vision Transformer): We utilize the ViT-base-patch16-224-in21k model Dosovitskiy et al. [2021], which has been pre-trained on a large image dataset. This model processes individual frames efficiently, extracting meaningful visual features relevant for frame-level understanding.
- ViViT (Video Vision Transformer): To capture temporal information across video sequences, we integrate the ViViT-B-16x2 model Arnab et al. [2021], pre-trained on the Kinetics-400 dataset. This model extends ViT’s capabilities by processing spatiotemporal patterns in videos, making it well-suited for analyzing surgical procedures.

These models ensure that both spatial and temporal features of surgical videos are effectively captured, providing a rich visual representation.

To encode textual information about detected surgical instruments and objects, we incorporate DistilBERT Sanh et al. [2019], a lightweight transformer model known for efficient text encoding. This ”distilled” version of BERT shows a good tradeoff between precision and efficiency.

For text generation, we adopt the T5 model. Specifically, we use:

- T5-Small: This variant is used for generating captions at the frame level.
- FLAN-T5-Base: This model better to generate coherent surgical clip-level captions.



The choice of T5 Raffel et al. [2020] is motivated by its strong performance in sequence-to-sequence tasks, enabling effective transformation of extracted features into human-readable surgical descriptions.

To refine and structure the final surgical report, several existing LLMs have been test, such as Med-Palm Singhal et al. [2023] developed by Google or BioGPT Luo et al. [2022] by Microsoft. However, while these fine-tuned models are specifically well-suited for medical tasks, their summarization competences are quite limited. Therefore, the employed LLM for this task is GPT-4 OpenAI [2023]. This model is particularly effective in summary generation, while maintaining a good knowledge in the medical field. It is also very efficient at adapting the output to follow structured reporting guidelines. After several tests, the retained prompt for this model is the following:

```
Generate a concise and textual surgery report from the following
sequential clip captions of a video. Each clip describes a phase of
the surgery, including the activity, tools used, and duration. **Key
Instructions:** 1. The clips form a continuous video. If multiple
clips describe the same activity, combine their durations to reflect
the total time spent on that activity. 2. Write the report in a narrative
format, explaining each phase step-by-step in a flowing text. Clip
captions: { clip captions}
```

By combining these models, we create a robust pipeline that extracts, processes, and generates meaningful descriptions of surgical procedures, ensuring both precision and clarity in automated reporting.

### 5.2.1 Clip caption generation scheme

The detailed generation workflow for a given clip can be seen in Figure 5.3. The inputs are the 32-frames and their corresponding description. Each modality is treated independently and brought into a latent space, and then fused and decoded thanks to the generation head.

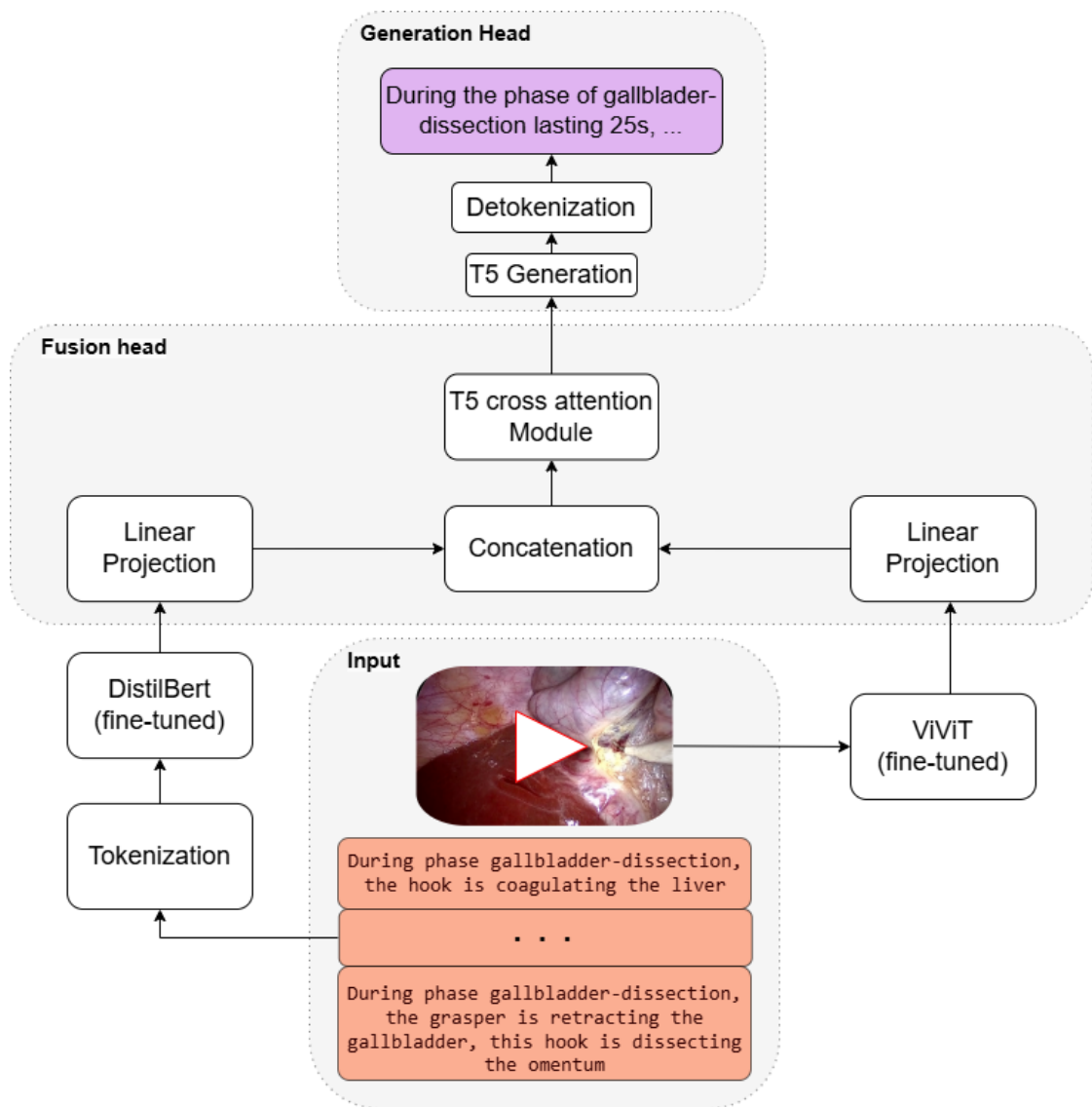


Figure 5.3: General scheme of the clip caption generation module

## 5.3 Evaluation Metrics

### 5.3.1 Classification metrics

To assess the performance of our classification model, we use four key metrics: Precision, Recall, F1 Score, and Accuracy. These metrics compare the model’s predictions against the ground truth, capturing different aspects of performance.

1. **Precision** measures the proportion of correctly predicted positive instances among all predicted positives, indicating reliability in positive classifications:

$$\text{Precision} = \frac{TP}{TP + FP}$$

2. **Recall** (Sensitivity) evaluates the model’s ability to identify all actual positives, ensuring minimal false negatives:

$$\text{Recall} = \frac{TP}{TP + FN}$$

3. **F1 Score** is the harmonic mean of precision and recall, balancing both metrics:

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4. **Accuracy** measures the overall proportion of correct predictions:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

These metrics provide a comprehensive evaluation: precision ensures correctness in positive predictions, recall captures all positive instances, F1 Score balances both, and accuracy reflects overall correctness.

### 5.3.2 Calibration

Expected Calibration Error (ECE) is a metric used to measure how well a model’s predicted probabilities align with the actual correctness of those predictions. For example, if a model predicts a class with 80% confidence, that class should be correct about 80% of the time.

Mathematically, ECE is computed by partitioning the predicted confidence values into  $M$  bins and then measuring the absolute difference between the average confidence and the accuracy within each bin. Let  $n$  be the total number of samples,  $\{B_m\}_{m=1}^M$  the set of bins. The following formula computes the ECE score:

$$ECE = \sum_{m=1}^M \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)| \quad (5.1)$$

One approach to improving calibration after training is temperature scaling. This method introduces a parameter  $T$  (temperature) that adjusts the model's logits before getting the predictions by applying the softmax function. The adjusted logits are computed as such:

$$\tilde{z} = \frac{z}{T} \quad (5.2)$$

where  $z$  represents the original logits and  $T > 1$  helps to smooth the predictions, reducing overconfidence.

The temperature parameter  $T$  is optimized using a small validation set by minimizing the negative log-likelihood:

$$T^* = \arg \min_T \sum_i -y_i \log \text{softmax} \left( \frac{z_i}{T} \right) \quad (5.3)$$

where  $y_i$  are the ground-truth labels and  $z_i$  are the original logits.

Once  $T^*$  is found, it is used during inference to scale the logits, leading to better-calibrated confidence scores without changing the model's classification accuracy.

### 5.3.3 Text Metrics

To assess the quality of the generated captions, we use widely recognized natural language evaluation metrics: BLEU and ROUGE. These metrics compare the model's generated text against reference captions, measuring aspects such as precision, recall, and sequence similarity.

1. **BLEU** evaluates how similar a generated caption is to a set of reference captions by computing the precision of  $n$ -grams (a sequence of  $n$  words) while incorporating a brevity penalty to discourage overly short outputs. The BLEU score for an  $n$ -gram order is computed as:

$$\text{BLEU}_n = BP \cdot \exp \left( \sum_{i=1}^n w_i \log p_i \right)$$

$$p_i = \frac{\sum_{\text{n-gram} \in \text{candidate}} C_{\text{clip}}(\text{n-gram})}{\sum_{\text{n-gram} \in \text{candidate}} C(\text{n-gram})}$$

where  $p_i$  is the precision of  $i$ -gram matches between generated and reference texts, and  $w_i$  are weighting factors.  $BP$  (brevity penalty) ensures that shorter outputs do not get disproportionately high scores:

$$BP = \begin{cases} 1, & \text{if } c > r \\ e^{(1-r/c)}, & \text{if } c \leq r \end{cases}$$

where  $c$  is the length of the generated caption and  $r$  is the length of the closest reference caption. BLEU outputs a score between 0 and 1, with higher scores indicating better alignment with the reference text.

2. **ROUGE** is a recall-based metric that measures how much of the reference text is captured in the generated output. We consider three key ROUGE variants:

- (a) ROUGE-1: Measures unigram (single word) recall, reflecting how many words in the reference appear in the generated text.

$$\text{ROUGE-1} = \frac{\# \text{ of overlapping unigrams}}{\text{total unigrams in reference}}$$

- (b) ROUGE-2: Extends ROUGE-1 to bigrams, capturing short phrase matches.

$$\text{ROUGE-2} = \frac{\# \text{ of overlapping bigrams}}{\text{total bigrams in reference}}$$

- (c) ROUGE-L: Uses the longest common subsequence (LCS) to measure the longest overlapping sequence of words, capturing fluency and coherence. It is computed as:

$$\text{ROUGE-L} = \frac{LCS(\text{generated}, \text{reference})}{\text{length of reference}}$$

where  $LCS$  denotes the longest sequence of words appearing in both texts in order.

3. **BERT** scores are also used to compute the resemblance between texts. In a similar way as in *Classification metrics*, the scores are divided in 3 components: Precision, Recall and F1 score. The formula for F1 score is the same as in 5.2.1. For precision and recall, the reasoning is the also the same, except *cosine\_similarity* is used between tokens. BERT scores capture how semantically similar the predicted text is to the reference text by comparing the contextual meaning of words rather than just exact matches. The formulas are defined as such:

$$\text{Precision} = \frac{1}{|\text{predicted tokens}|} \sum_{t \in \text{predicted tokens}} \max_{r \in \text{reference tokens}} \text{cosine\_similarity}(t, r)$$

$$\text{Recall} = \frac{1}{|\text{reference tokens}|} \sum_{r \in \text{reference tokens}} \max_{t \in \text{predicted tokens}} \text{cosine\_similarity}(r, t)$$

These metrics collectively provide a comprehensive evaluation of the generated captions, assessing their lexical accuracy (BLEU), factual consistency (ROUGE-1 and ROUGE-2), structural similarity (ROUGE-L), and semantic alignment (BERT).

## 5.4 Training Strategy

The step by step clip caption generation  $Objects \rightarrow Frame\ Caption \rightarrow Clip\ Caption$  benefits from the fact that each model is specialized in a specific task, using as input the output as the model before. However, models can still make mistakes. To improve the robustness of the models, we train the Frame Captioner ( $FC$ ) to adapt to the Object Detector model ( $OD$ ) errors. Similarly, the Clip Captioner ( $CC$ ) is adapted to learn from  $FC$ 's mistakes. In other words, considering input image  $I$  and clip  $Clip$  with the corresponding ground truth labels  $O$ ,  $F$ ,  $C$ , the training is divided in the two following parts.

1.

$$\tilde{O} = OD(I), \tilde{F} = FC(I, O), \tilde{C} = CC(Clip, F)$$

In this part, each of the 3 models is trained independently from the others.

2.

$$\hat{F} = FC(I, \tilde{O}), \hat{C} = CC(clip, \hat{F})$$

Now, instead of taking as input the ground truth objects  $O$ ,  $FC$  takes as input the output of the object detection model. The same goes for the clip caption generation.

## 5.5 Resources and technical specifications

The original *CholecT50* CAMMA [2020] is 59Gb, and the processed dataset for frame- and clip-level analysis are respectively 50.44 Gb and 111.86 Go. Due to the large size of the clip-level dataset, the training was done sequentially by separating the dataset in 2 and training one after an other using checkpoints. The GPU used was A100, offering 83 Gb system RAM, 40 Gb GPU RAM and 235 Gb storage. The training of the heaviest model; the robust clip caption generator; lasted approximately 7 hours, with around 30 minutes per epoch. This slow training is due to the large amount of trainable parameters; the details for each model can be found here 7.

## Chapter 6

# Experiments

The following section presents the results of applying our methodology to our dataset, with the specificities explained in the previous section. The first part displays an example of the process for a given clip, while we focus on numerical results on the second part.

### 6.1 Workflow Example

The following figure 6.1 displays a complete example of the analysis process of the first clip of the first video: *video* = VID01, *frames* = 000000 – 000031. The grasp and gallbladder are immediately detected on the first frames. The model detects this corresponds to the preparation phase as we can see on the first frame captions. Then, the surgeon starts using a hook to begin the carlot triangle dissection phase.

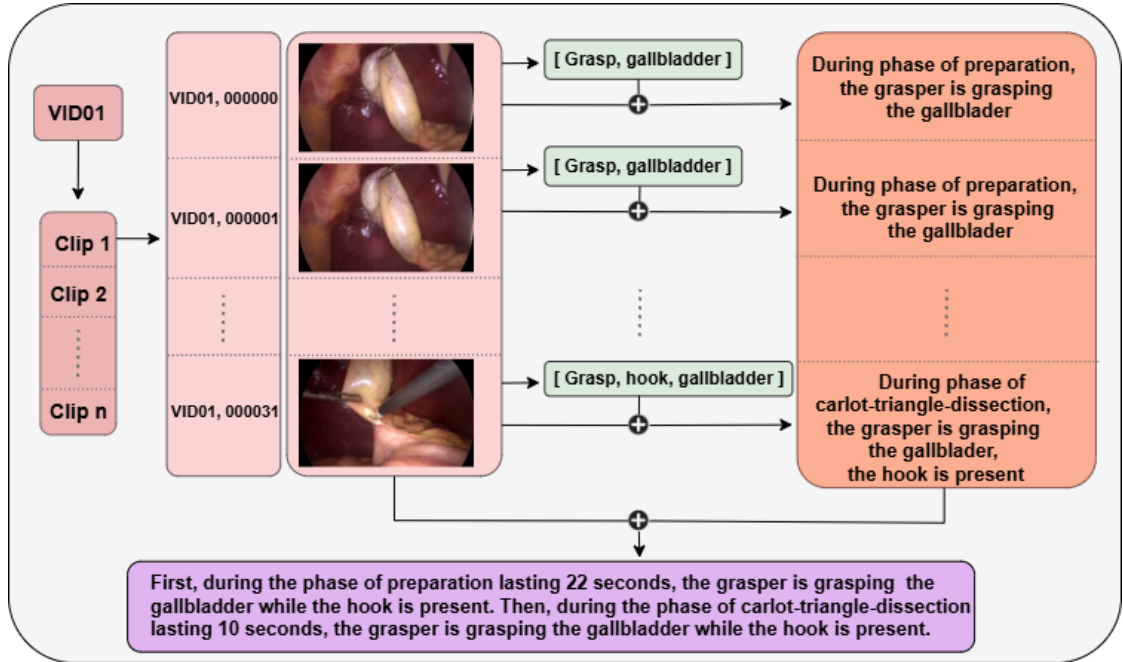


Figure 6.1: Workflow Example

## 6.2 Results

### 6.2.1 Object Detection

The model is highly precise, correctly identifying relevant instances 96.17% of the time. It also has strong recall, capturing 95.24% of all relevant cases. The F1 score, which balances precision and recall, is 95.70%, indicating a well-rounded performance. Overall, the model’s accuracy is 83.11%, meaning it makes correct predictions in the majority of cases. Table 6.1 displays the specific statistics for each different tool or target. Excluding *specimen bag* which has null statistics because there is no instance of it in the dataset, we notice the poorest results for *abdominal\_wall\_cavity* with a precision of 66.67%. This may be due to its lack of clear edges and distinct features, blending into surrounding tissues.

Label Name	Precision	Recall	F1 Score
grasper	0.9839	0.9673	0.9755
bipolar	0.9737	0.8998	0.9353
hook	0.9627	0.9893	0.9758
scissors	0.9005	0.8873	0.8938
clipper	0.9329	0.8870	0.9094
irrigator	0.9476	0.9163	0.9317
specimen_bag	0.0000	0.0000	0.0000
gallbladder	0.9758	0.9706	0.9732
cystic_plate	0.8643	0.9370	0.8992
cystic_duct	0.9066	0.8971	0.9018
cystic_artery	0.9248	0.8120	0.8647
cystic_pedicle	0.9474	0.6667	0.7826
blood_vessel	0.9677	0.7692	0.8571
fluid	0.8638	0.8142	0.8383
abdominal_wall_cavity	0.6667	0.7105	0.6879
liver	0.9488	0.9273	0.9380
adhesion	0.9500	0.8636	0.9048
omentum	0.9762	0.9167	0.9455
peritoneum	0.9333	0.8829	0.9074
gut	0.9783	0.6338	0.7692
specimen_bag	0.8836	0.9000	0.8917

Table 6.1: Label-wise Precision, Recall, and F1 Score

The ECE (Expected Calibration Error) before calibration reached 0.0075, which already indicates a well-calibrated model. However, by applying temperature scaling, the ECE improved further to 0.0028, showing that the model became even more confidently aligned with its predictions after calibration. Although the initial ECE was good, the temperature scaling helped refine the model’s confidence, resulting in better-calibrated probabilities. The optimal temperature of  $T = 1.8584$ , the procedure softened the confidence score and reduced the overconfidence of the model. This improvement can be seen in Figure 6.2, where the plots before and after calibration highlight this improvement. These results have been made by keeping a threshold of 0.5, because it should be the best performance in terms of calibration and precision.



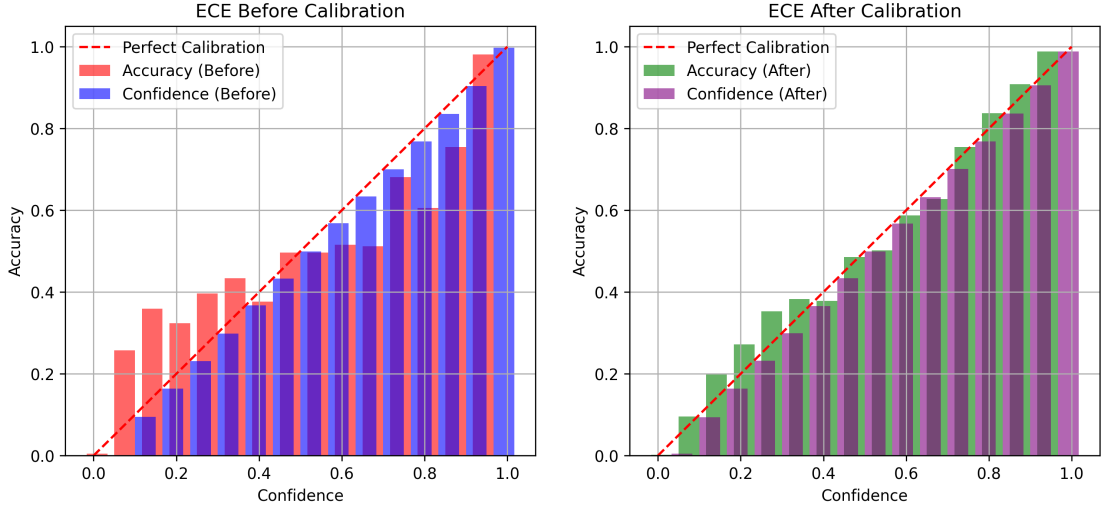


Figure 6.2: ECE plots of Object detection model

### 6.2.2 Frame Caption Generation

The following table 6.2 shows the different statistics for both the Initial model and the robust model. As mentioned in section 5.4, the robust model is a fine tuned version of the original Frame caption generator, by taking the predicted objects instead of ground-truth objects as input. Non surprisingly, learning to compensate the Object detector mistakes upgrade the models performance, going from 0.64 to 0.73 with the *BLEU* score. We also notice a better performance when it comes to *ROUGE* statistics. However, the *BERT* precision remains similar for both models. This can explain itself by the fact that *BERT* measure the semantic similarity between predicted and target captions, and even the original model is able to generate words semantically close to the target caption. We still observe an improvement for the *Recall* and *F1* going respectively from 0.76 and 0.77 to 0.84 and 0.80. Examples of generated frame captions from the robust model can be found in appendix 7.

Metric	Model	Robust Model
BLEU	0.6395	0.7267
ROUGE-1	0.8351	0.8700
ROUGE-2	0.7747	0.8096
ROUGE-L	0.8116	0.8637
BERT Precision	0.7771	0.7745
BERT Recall	0.7644	0.8365
BERT F1	0.7707	0.8052

Table 6.2: Performance comparaison

### 6.2.3 Clip caption Generation

Different models have been tested for the clip caption generation. The following table 6.3 presents their results. The *Model* and *Robust* represent the 2 models

presented in section 5.4. *Simple* model is a lightened version is the unimodal version of the model, taking only the video as input and not considering the frame captions. It was implemented to analyze the impact of taking the frame captions as an input. It is relatively straightforward that this unimodal version of the model has poorer results as the other ones with for example a BLEU score of 0.51 and BERT precision of 0.66. The difference of the *Model* by taking ground-truth frame captions and the created ones by the frame caption generator is also observable. Indeed, with at least 0.02 points less for every statistics, the model taking as input the generated captions did not learn from the possible mistakes already in them. Furthermore, we observe a bigger gap between both models when it comes to BERT precision and F1. However, the *Robust* model surpasses the measurements of every other model, highlighting the need to train the object detector, frame- and clip-generator both independently first and dependently on a second time. This final mode became both more confident and more precise, reaching a *ROUGE – L* score of 0.83.

Frame captions input type	None	Ground-Truth	Generated	
Model type	Simple	Model	Model	Robust
BLEU	0.5138	0.6490	0.6023	0.6715
ROUGE-1	0.7591	0.8615	0.8317	0.8672
ROUGE-2	0.6744	0.7975	0.7447	0.7991
ROUGE-L	0.7137	0.7968	0.7605	0.8318
BERT Precision	0.6666	0.7733	0.6705	0.7443
BERT Recall	0.6590	0.7696	0.7486	0.7772
BERT F1	0.6623	0.7714	0.7090	0.7607

Table 6.3: Comparison of model performance depending on training strategy and input type

## Chapter 7

# Conclusion

The approach presented in this study shows promising results in the framework of surgery video analysis. Indeed, by combining the specialties of each model, respectively, object detection, visual feature extraction, and temporal features extraction, we limit the visual hallucinations often present in computer vision models. Also, the results showed that the multi-turn reasoning helped the model to capture the overall context of the videos with an improvement of up to 12% for semantic precision and 30% for the BLEU score.

The promising results are yet to be improved in several ways. First, a more complete dataset would enable the models to cover different types of surgeries and enlarge the possibilities of application of the model, and turn this approach into a medicine-specific foundation model. Secondly, the selected video vision transformer processes frames with a span of 2 seconds. The temporal window of the patches could be expanded to 4 or 8 seconds, allowing the model to capture dependencies on a longer range. Finally, we have seen the strength of this approach resides in the different modalities and the multi-turn reasoning. Indeed, By dividing the main objective into different sub-tasks with a specialized agent for each of them, we reduce the number of possible errors. One could consider additional modalities before the frame caption generation. For instance, considering the objects' position thanks to segmentation, or the sound from surgical tools could provide a richer context for more accurate and informative captions.

It is important to note that the models have here been trained on the same dataset. Even though resources are quite limited for surgery videos, it is still possible to train the different modalities on different datasets depending on the annotations available. This practice would make the combination of the models more general and able to cover different types of procedures. Of course, the need of precision is specifically important in the realm of medicine, but the global architecture could be extended to other disciplines such as autonomous driving or security monitoring.

# Bibliography

- Daniel A. Hashimoto, Guy Rosman, Daniela Rus, and Ozanan R. Meireles. Artificial intelligence in surgery: Promises and perils. *Annals of Surgery*, 268(1):70–76, July 2018. doi: 10.1097/SLA.0000000000002693. URL <https://pmc.ncbi.nlm.nih.gov/articles/PMC5995666/>.
- Yicheng Feng, Yijiang Li, Wanpeng Zhang, Sipeng Zheng, and Zongqing Lu. Videorion: Tokenizing object dynamics in videos, 2024. URL <https://arxiv.org/abs/2411.16156>.
- Long Zhao, Nitesh B. Gundavarapu, Liangzhe Yuan, Hao Zhou, Shen Yan, Jennifer J. Sun, Luke Friedman, Rui Qian, Tobias Weyand, Yue Zhao, Rachel Hornung, Florian Schroff, Ming-Hsuan Yang, David A. Ross, Huisheng Wang, Hartwig Adam, Mikhail Sirotenko, Ting Liu, and Boqing Gong. Videoprism: A foundational visual encoder for video understanding, 2024. URL <https://arxiv.org/abs/2402.13217>.
- Shehan Munasinghe, Hanan Gani, Wenqi Zhu, Jiale Cao, Eric Xing, Fahad Shahbaz Khan, and Salman Khan. Videoglamm: A large multimodal model for pixel-level visual grounding in videos, 2025. URL <https://arxiv.org/abs/2411.04923>.
- Peng Xu, Wenqi Shao, Kaipeng Zhang, Peng Gao, Shuo Liu, Meng Lei, Fanqing Meng, Siyuan Huang, Yu Qiao, and Ping Luo. Lvlm-ehub: A comprehensive evaluation benchmark for large vision-language models, 2023. URL <https://arxiv.org/abs/2306.09265>.
- Sahar Nasirihaghighi, Negin Ghamsarian, Heinrich Husslein, and Klaus Schoeffmann. Event recognition in laparoscopic gynecology videos with hybrid transformers, 2023. URL <https://arxiv.org/abs/2312.00593>.
- Dimitri Kiyasseh, Rui Ma, Taimur F. Haque, Benjamin J. Miles, Christian Wagner, David A. Donoho, Anima Anandkumar, and Jason A. Hung. A vision transformer for decoding surgeon activity from surgical videos, 2023. URL <https://doi.org/10.1038/s41551-023-01010-8>.
- Mikhail Volkov, Daniel A. Hashimoto, Guy Rosman, Ozanan R. Meireles, and Daniela Rus. Machine learning and coresets for automated real-time video segmentation of laparoscopic and robot-assisted surgery. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 754–759, 2017. doi: 10.1109/ICRA.2017.7989093.

- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2023. URL <https://arxiv.org/abs/1706.03762>.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xi-aohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL <https://arxiv.org/abs/2010.11929>.
- CAMMA. Cholect50: A dataset for surgical video understanding, 2020. URL <https://github.com/CAMMA-public/cholect50>. Accessed: 2025-03-08.
- Anurag Arnab et al. Vivit: A video vision transformer. *ICCV*, 2021.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS 2019*, 2019.
- Colin Raffel et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020.
- Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, Mike Schaekermann, Amy Wang, Mohamed Amin, Sami Lachgar, Philip Mansfield, Sushant Prakash, Bradley Green, Ewa Dominowska, Blaise Aguerre y Arcas, Nenad Tomasev, Yun Liu, Renee Wong, Christopher Sementurs, S. Sara Mahdavi, Joelle Barral, Dale Webster, Greg S. Corrado, Yossi Matias, Shekoofeh Azizi, Alan Karthikesalingam, and Vivek Natarajan. Towards expert-level medical question answering with large language models, 2023. URL <https://arxiv.org/abs/2305.09617>.
- Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. BioGPT: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6), 09 2022. ISSN 1477-4054. doi: 10.1093/bib/bbac409. URL <https://doi.org/10.1093/bib/bbac409>. bbac409.
- OpenAI. Gpt-4 technical report, 2023. URL <https://openai.com/research/gpt-4>.
- Riccardo Cantini, Cristian Cosentino, and Fabrizio Marozzo. Multi-dimensional classification on social media data for detailed reporting with large language models. In Ilias Maglogiannis, Lazaros Iliadis, John Macintyre, Markos Avlonitis, and Antonios Papaleonidas, editors, *Artificial Intelligence Applications and Innovations*, pages 100–114, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-63215-0.
- Riccardo Cantini, Cristian Cosentino, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Harnessing prompt-based large language models for disaster

monitoring and automated reporting from social media feedback. *Online Social Networks and Media*, 2025a. URL <https://api.semanticscholar.org/CorpusID:274297218>.

Riccardo Cantini, Cristian Cosentino, Irene Kilanioti, Fabrizio Marozzo, and Domenico Talia. Unmasking deception: a topic-oriented multimodal approach to uncover false information on social media. *Machine Learning*, 114(1):13, 2025b. doi: 10.1007/s10994-024-06727-4. URL <https://link.springer.com/article/10.1007/s10994-024-06727-4>.

# Appendix

## Frame Captions

In this section are presented some generated caption examples. Despite the positive results presented in section 6.2, the model has trouble generating the word *gallbladder* and sometimes generates "*gallbloddger*" instead as in the second example. This issue is likely due to tokenization, where the model may incorrectly split or merge tokens, leading to these small mistakes.

---

**Predicted:** During phase carlot-triangle-dissection, the grasper is grasping the gallbladder, their bipolar is dissecting the cystic\_artery

**Target:** During phase carlot-triangle-dissection, the bipolar is dissecting the cystic\_artery, the grasper is grasping the gallbladder

---

**Predicted:** During phase gallbladder-dissection, the hook is dissecting the gallbloddger

**Target:** During phase gallbladder-dissection, the hook is dissecting the gallbladder

---

**Predicted:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, this hook is present

**Target:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, the hook is present

---

**Predicted:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, those bipolar is coagulating the abdominal\_wall\_cavity

**Target:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, the bipolar is coagulating the abdominal\_wall\_cavity

---

**Predicted:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, this hook is dissecting the omentum

**Target:** During phase carlot-triangle-dissection, the grasper is retracting the gallbladder, the hook is dissecting the omentum

## Report Example

Report for VID07:

---

The surgical procedure initiated with the preparation phase, which lasted for a total of 56 seconds. Initially, the grasper was engaged in retracting the gallbladder while the hook was used to dissect the gallbladder. Following this, the clipper retracted the omentum for 8 seconds. The next phase was the Calot triangle dissection, which took an extended duration due to the complexity involved. The hook was used to dissect the gallbladder while the grasper helped in retracting the gallbladder. The dissection of cystic artery and cystic duct was carried out multiple times during this phase. The grasper maintained a consistent presence, assisting in retracting the gallbladder throughout. The procedure moved on to the clipping and cutting phase which lasted for 117 seconds in total. The grasper was utilised to retract the gallbladder while the clipper was engaged in clipping the cystic artery and the cystic duct. The scissors were also used to cut the cystic duct and artery. Following this, the gallbladder dissection phase was carried out for a prolonged duration. The hook dissected the gallbladder, and the grasper retracted the gallbladder and the liver simultaneously. During this phase, the irrigator was also used to retract the liver while the grasper was present. The gallbladder packaging phase took 48 seconds. The grasper was engaged in packing the gallbladder, and the liver was retracted. The specimen bag was handled by the grasper during this phase. The cleaning and coagulation phase took place next. The grasper was used to retract the liver, the irrigator was employed to aspirate the fluid, and the bipolar was used to coagulate the omentum. There were instances where the omentum was aspirated and the abdominal wall cavity was irrigated. The final phase was the gallbladder extraction, which lasted for 79 seconds. The grasper was present throughout, the hook was used for gallbladder dissection, and the scissors were used for handling the specimen bag. The procedure concluded with the extraction of the gallbladder.



## Models Parameters

### Object Detector

Layer (type)	Output Shape	Param #
ViTModel-1	[-1, 196, 768]	85207296
Linear-2	[-1, 21]	16149

Total params: 85223445  
Trainable params: 85223445  
Non-trainable params: 0

### Frame Captioner

Layer (type)	Output Shape	Param #
ViTModel-1	[-1, 196, 768]	86389248
Linear-2	[-1, 196, 512]	393728
DistilBertModel-3	[-1, 64, 768]	66362880
Linear-4	[-1, 64, 512]	393728
Fusion-5	[-1, 260, 512]	0
T5ForConditionalGeneration-6	[-1, num.tokens, 32128]	60506624

Total params: 214046208  
Trainable params: 214046208  
Non-trainable params: 0

### Clip Captioner

Layer (type)	Output Shape	Param #
VivitModel-1	[-1, 3137, 768]	89236992
Linear-2	[-1, 3137, 768]	590592
DistilBertModel-3	[-1, 4096, 768]	66362880
Linear-4	[-1, 4096, 768]	590592
Fusion-5	[-1, 7233, 768]	0
T5ForConditionalGeneration-6	[-1, num.tokens, 32128]	248168448

Total params: 404358912

Trainable params: 404358912  
Non-trainable params: 0

---

**Declaration of originality**

The signed declaration of originality is a component of every written paper or thesis authored during the course of studies. In consultation with the supervisor, one of the following three options must be selected:

- ☐ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used no generative artificial intelligence technologies<sup>1</sup>.
- ☒ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used and cited generative artificial intelligence technologies<sup>2</sup>.
- ☐ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used generative artificial intelligence technologies<sup>3</sup>. In consultation with the supervisor, I did not cite them.

**Title of paper or thesis:**

A new multimodal approach for surgery Video report generation

**Authored by:**

*If the work was compiled in a group, the names of all authors are required.*

**Last name(s):**

Georgenthum

**First name(s):**

Hugo

With my signature I confirm the following:

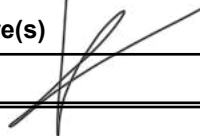
- I have adhered to the rules set out in the Citation Guide.
- I have documented all methods, data and processes truthfully and fully.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for originality.

**Place, date**

Zürich, 16/03/2025

**Signature(s)**



*If the work was compiled in a group, the names of all authors are required. Through their signatures they vouch jointly for the entire content of the written work.*

<sup>1</sup> E.g. ChatGPT, DALL E 2, Google Bard

<sup>2</sup> E.g. ChatGPT, DALL E 2, Google Bard

<sup>3</sup> E.g. ChatGPT, DALL E 2, Google Bard